Technical Document 2036
January 1991

Graph Coloring Applied to Voice Conferencing in Tactical Packet Radio Networks

N. Davé

NAVAL OCEAN SYSTEMS CENTER
San Diego, California 92152-5000

J. D. FONTANA, CAPT, USN
Commander

H. R. TALKINGTON, Acting
Technical Director

ADMINISTRATIVE INFORMATION

This work was performed by the System Design and Architecture Branch, Code 854, Naval Ocean Systems Center, for the Chief of Naval Research, OCNR-10P, Independent Exploratory Research program.

Released by
R. L. Merk, Head
System Design and
Architecture Branch

Under authority of
K. R. Casey, Head
Battle Force and
Theater Communications
Division

ACKNOWLEDGMENTS

Major funding for the work reported herein was provided by the Office of Naval Research, Independent Research projects, through Dr. Alan Gordon of Naval Ocean Systems Center (Job Order No. ZW30850A01). Additional funding was provided by the Shared Adaptive Internetworking Technology (SAINT) project (Job Order No. CHB6850W01).

JG

SUMMARY

Several algorithms for increasing the data rate in an ultrahigh frequency line-of-sight (UHF-LOS) network for the purpose of voice conferencing are investigated. These are caller-controlled routing/time-slotting (CRS), distributed routing/time-slotting (DRS), distributed entire network slotting and reslotting (DENSR), and distributed entire network slotting and reslotting with topology updating (DENSRTU).

The above algorithms were tested in a simulation testbed under changing network topologies (network dynamics). Two models of network dynamics were studied: Model 1, in which initially existing links are randomly broken and reestablished; and Model 2, in which two nodes are allowed to move relative to the remainder of the network, which remains stationary.

RESULTS

The DENSR algorithm is found, in general, to be the most reliable algorithm for generating successful voice conferences under the network dynamics studied. However, if the rate of link changes is low and the degree c of the network (degree of the most connected node or nodes in the network) approaches the number of nodes n in the network, this algorithm may not yield any improvement in effective data rates, and the overhead required to execute it may be ill-spent. In such a case the CRS algorithm, with its economy of conference duration time and voice slots per cycle (s) and sparseness of relays used in the conference circuit, may be preferable. Evidence of these virtues of the CRS algorithm relative to the DENSR algorithm is shown in the simulations on Model 2 of topology dynamics.

RECOMMENDATIONS

The DENSR algorithm is, in general, to be preferred on the basis of reliability (success rate in generating voice conferences). However, for high-degree, low-radius networks the CRS algorithm may be preferable. Thus, a decision-making algorithm, which chooses between the DENSR and CRS algorithms based on network degree, is recommended.

CONTENTS

FIGURES

CONTENTS (continued)

FIGURES (continued)

CONTENTS (continued)

FIGURES (continued)

1.0 BACKGROUND

Tactical packet radio networks normally obey the assigned slot time division multiple access (TDMA) channel access protocol as described, for example, by Tannenbaum (1988), wherein each node has one transmission slot, uniquely assigned to it in a "round robin" fashion, in which to transmit a "packet" of data containing a given number of bits. A TDMA "net cycle" or "epoch" consists of the sequence of slots assigned to the nodes in the network in the order that they arise in time (no gaps in time exist wherein some node is not assigned a slot). Thus, if the network contains n nodes, there are n slots in a TDMA net cycle. Such networks are of the "store-and-forward" type, meaning that messages are buffered up at a node until the node's transmission slot time arrives and a protocol is in place to determine which message is at the top of the forward buffer. The physical medium of communication for a tactical radio network may be the ultrahigh frequency line-of-sight (UHF-LOS) channel, the high-frequency (HF) skywave or groundwave channel, or a satellite channel. The UFH-LOS medium is the assumed medium of interest for this work. With this medium, connectivity is limited to those nodes lying within the transmitting node's line of sight, or within about 15 nautical miles (nmi) of the transmitter, so that nodes more distant can be reached only by relaying packets between intermediate nodes. Current UHF-LOS technology on naval platforms is limited in data rate to about 4800 bps; this limitation is due mostly to interference introduced into transmissions by the platform's superstructure. However, data rates of up to 14,400 bps may be possible in the future through refinement of technology.

Because naval platforms are mobile and because both hostile and friendly jamming are definite threats, the ability of two nodes to hear one another (known as the network "connectivity" or "topology") can change unpredictably with time, i.e., the topology is a dynamic variable. A node can only learn of the current connectivity of distant parts of the network (i.e., of nodes that are not its neighbors) by receiving messages from other nodes, which include as one of their message fields lists of identifiers of nodes whom those nodes can hear and who hear them. As a consequence, a node's knowledge of the network's topology may be outdated or incomplete since, depending on the message buffer queue service protocol that is in place, not all messages in a store-and-forward network are necessarily forwarded.

A variant of the TDMA protocol in which each node is assigned two TDMA slots in succession has been proposed for tactical data networks by Norvell, Brown, and Vineberg (1985). This variant has an adaptive channel access protocol under which a node can "handoff" one or both of its slots to a neighbor for use in case the neighbor's stored messages' buffer lengths indicate it needs the slots more than the slots' owner does. This protocol, known as handoff assigned multiple access (HAMA), is designed to increase the number of messages accommodated by the network compared

to a strict TDMA scheme, in which a node's bandwidth may either be wasted if it has no message to transmit or be inadequate compared to the number of messages waiting in its forward buffer. A penalty paid for this increased performance is that a net cycle under the HAMA protocol is twice as long as under TDMA, assuming the same slot size, i.e., for an n node network, $2n$ slots are required. Thus, a node under the HAMA protocol has to wait twice as long as one under TDMA for its slots to arise, although it has assigned to it twice as many slots per cycle as a node under TDMA (i.e., same number of slots per unit time). A sample 24-node network is shown in figure 1a, and time lines corresponding to one net cycle under TDMA and under HAMA channel access protocols are shown in parts b and c of this figure.

For a network of n nodes using the assigned TDMA channel access protocol with transmission time slots of duration t, a given node can transmit only once in the interval of time ($n\ t$), so that if the physical medium supports a data rate (bandwidth) of D_{med}, the effective data rate under assigned TDMA channel access protocol, D_{TDMA}, is

$$D_{\text{TDMA}} = \frac{D_{\text{med}}\, t}{n\, t} = \frac{D_{\text{med}}}{n} . \tag{1a}$$

Since multiple messages may be received by a given node in a network over the course of a net cycle, and the node may also generate its own messages, decisions have to be made concerning which message is to be relayed when the node's assigned TDMA slot arises and which are to be queued up in the node's message buffer. These decisions, called queue-serving protocols, may cause some messages to be delayed for a relatively long time at a node and even to be destroyed if they are too old, i.e., if a message "time-to-live" protocol is operational. Therefore, in an assigned TDMA scheme, since each node can transmit only once during the net cycle, the minimum delay of a packet at each relay node averages one-half of a net cycle and can be much longer. Consequently, a packet may require an indeterminate time to reach its destination, or may not reach its destination at all. In light of this discussion, equation 1a is actually an upper bound on the effective data rate between two nodes that are not neighbors in a network under assigned TDMA, since some messages may not be forwarded at all.

For many types of message traffic in tactical situations, failure to forward (destruction) of a small fraction of messages is not necessarily critical. For example, a large proportion of the messages being relayed by a node in a tactical radio network may be routine "track" information records from various nodes' radar. Since these may be generated on a regular basis, it may not be important for every member of the network to receive all of the track records from every node. Thus, indeterminate time delays in receiving relayed messages may not be of great consequence for many types of message traffic in view of a major advantage offered by the TDMA and HAMA channel

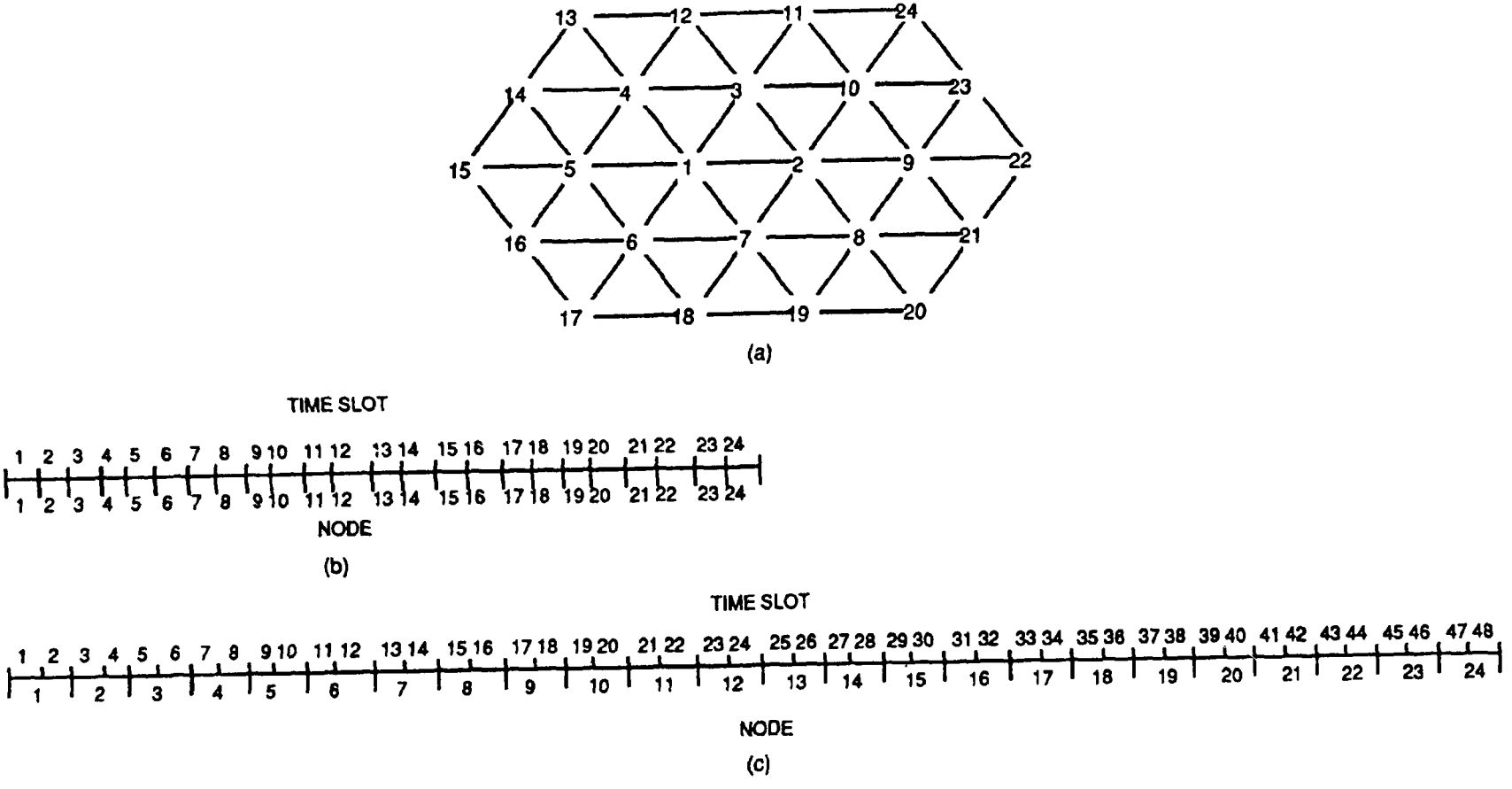


Figure 1. (a) Topology of a sample 24-node network. (b) Time line showing slot assignments for each node using an assigned slot time division multiple access (TDMA) channel access protocol. (c) Time line showing slot assignments for each node using a handoff assigned multiple access (HAMA) channel access protocol.

access protocols: each node in the network is guaranteed a certain amount of time during a net cycle (bandwidth), or a certain fraction of the channel, to transmit data. Evidently, from equation 1a, the fraction F_{TDMA} of the total bandwidth of the medium available to a given node under assigned TDMA, which may also be called the TDMA "channel utilization fraction," is given by

$$F_{TDMA} \equiv \frac{D_{TDMA}}{D_{med}} = \frac{1}{n} . \quad (1b)$$

For other types of message traffic, however, the low value of F_{TDMA} of equation 1b and the delays inherent in TDMA or HAMA may be annoyingly long or even intolerable. A prime example of the need for accelerated data flow rates and guaranteed delivery of packets between some nodes in a network is provided by the case of high-priority packetized voice traffic—either between a caller and callee (point-to-point voice) or between a caller and several callees (voice conferencing). Such communication in a UHF-LOS network generally requires that voice packets be relayed on a "voice circuit" from caller to callees (since callees generally are not neighbors of the caller), and that nodes neighboring voice circuit members cease all transmissions, or be in a "passive" state so as not to interfere with the voice traffic until such traffic has ceased (Gutman, Casey, Merk, Olsen, and Warner, 1988). For voice transmissions to be comprehensible, the data must arrive above a certain minimum rate, which is generally considered to be about 1000 bits per second (bps), with higher data rates yielding better voice quality and less susceptibility to environmental degradations (Gold, 1977). Below this value of data transmission rate, voice transmissions are only marginally intelligible. For a TDMA network of n nodes, assuming that all nodes forward only the caller's voice packets and no other data, equation 1a shows that the caller would have to transmit (1000 n) bits per second of voice traffic to maintain an intelligible voice data rate. If the network consists of up to 10 nodes, such data rates (on the order of 10,000 bps) might be achievable within a TDMA channel access protocol for a UHF-LOS network, which as pointed out earlier is limited to approximately such data rates with currently available or slightly enhanced technology, but in networks consisting of more nodes such a data rate requirement would clearly be prohibitive. An alternative is to buffer up received voice packets at destination nodes and to resynthesize or play back the voice after enough packets have been received, but this procedure also involves potentially long delays. Thus, for successful voice traffic or conferencing in a UHF-LOS network, it is necessary to modify the usual TDMA channel access protocol.

Merk, Gutman, and Warner (1988) and Gutman, et al. (1988) outline a solution of this problem for point-to-point voice. In their proposed method, a voice circuit consisting of two participants (caller and callee) and relays by which one may transmit to the other is created in a packet radio relay network so a mere three-slot net cycle is used

by the nodes on the circuit; neighbor nodes of circuit members remain silent or passive during the time that the participant nodes carry voice traffic. By the same argument used in obtaining equation 1, it is clear that the effective data rate of this point-to-point voice circuit, D_{pp} is

$$D_{pp} = \frac{D_{\text{med}}}{3} . \tag{2a}$$

so the fraction F_{pp} of the channel available to the caller (or callee) for one-way, point-to-point voice traffic is

$$F_{pp} \equiv \frac{D_{\text{med}}}{D_{pp}} = \frac{1}{3} . \tag{2b}$$

Thus, for a point-to-point voice call, a tactical radio network is capable of providing the minimum data rate given by equation 2a, which should give adequate intelligibility of voice in a 4800-bps data rate UFH-LOS network.

In the context of the more general problem of voice conferencing, i.e., voice communication between a caller and more than one callee, it is of interest to inquire how the flow of message traffic between at least a few chosen nodes, if not the entire network, may be accelerated. It is clear that without increased data transmission rates during a node's assigned slot and without departing from an assigned transmission slot protocol, which involves slot cycles for channel access, data rates can only be speeded up if it is possible for more than one node to be assigned the same time slot, in analogy with the point-to-point voice scheme discussed above. With more than one node sharing a time slot, the total number of slots per cycle could be some integer s lying between the minimum of three slots required for point-to-point voice and the maximum of n slots for TDMA:

$$3 \leq s \leq n \tag{3}$$

with effective data rate, D_{vc} , and channel utilization fraction F_{vc} given by

$$D_{vc} = \frac{D_{\text{med}}}{s} , \tag{4a}$$

$$D_{vc} = \frac{D_{\text{MED}}}{D_{\text{med}}} = \frac{1}{s} . \tag{4b}$$

(In equation 3, and in this work, we ignore the trivial case in which a point-to-point call is between two neighbors in a network, in which the minimum for s is two.)

The smaller the value of s that is possible to use, the higher the effective data rate between caller and callees and the better the quality of the voice transmission.

Such a "slot reassignment" protocol (a summary of which is given by Gutman, et al. (1988)) falls within the mathematical discipline of graph theory, specifically graph coloring theory, one of the best known references for which is perhaps Christofides (1975). Graph coloring theory may be used to determine the minimum number of slots required for a given topology in an assigned time slot cycle to assure that transmissions from two nodes will not interfere with each other. Were it not for the fact that the connectivity of the tactical network is a dynamic variable, the problem of "time-slotting," or more appropriately "reslotting," the network (since the normal TDMA slot assignment can be viewed as a time-slotting, albeit generally not optimal) would be a fairly straightforward application of graph coloring theory. Because of the variability of tactical data network topology, however, no single node has perfect knowledge of network topology at any time, implying that packet routing and subnetwork reslotting decisions made at any one node will be prone to errors, which can cause failures in relaying a transmission such as a voice message. Thus, such "caller-controlled" routing and coloring algorithms, though perfectly appropriate for static topologies of which every node has perfect knowledge, would be expected to give only mediocre performance in dynamic topologies.

In the course of the work reported herein, therefore, a number of algorithms are developed, based on graph coloring theory, to accelerate the flow of data in networks obeying TDMA or HAMA type channel access protocols and having dynamic topologies. Specifically, the theory is used in a computer-based simulation to test the feasibility, in the face of changing network topology, of "time-slotting" a network (or a subnetwork of the original network) to operate with fewer slots per net cycle than the original TDMA or HAMA scheme, thus providing voice or other higher data rate transmission capability between a selected subset of nodes of the network, with reversion to the normal TDMA (HAMA) channel access protocol when the transmission is completed.

Section 2.0 examines how graph coloring theory can be used to determine which nodes may share the same slot in a network. A simple formula, in the Optimal Time-slotting Theorem, is derived for determining a lower bound on the number of time slots required in a net cycle for a given topology. In section 3.0, details are given of several algorithms that use the theory developed in section 2.0 to recolor the tactical TDMA network and to set up a network or subnetwork using fewer than n slots per net cycle, where n is the number of nodes in the network. The algorithms discussed are named as follows:

a. Caller-controlled routing/time-slotting (CRS)

b. Distributed routing/time-slotting (DRS)

c. Distributed entire network slotting and reslotting (DENSR)

d. Distributed entire network slotting and reslotting with topology updating (DENSRTU).

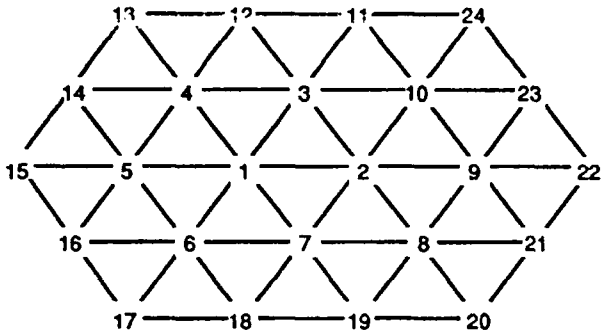
The implications of these names are explained in section 3.0. In section 4.0 we discuss the testing of these algorithms in an ADA language simulation program for tactical data networks (Mumm and Ollerton, 1990), including modeling of network topology dynamics. The last section discusses the results of the simulation for different rates of network topology changes and the implications of the work for improved data rates in tactical TDMA networks for packetized voice transmission and general applications.

2.0 GRAPH COLORING THEORY APPLIED TO TIME SLOT ASSIGNMENTS IN A NETWORK

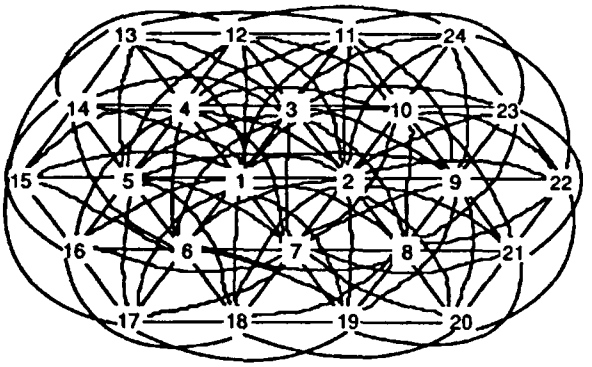
Graph theory (Christofides, 1975) is concerned with points called "nodes" or "vertices" and the connections between them, called "links." A graph may contain some links, which exist only in one direction, say from node i to node j but not in the reverse direction. (We will use "i" to denote "node i" where no possibility of confusion exists.) Such graphs are called directed graphs. In the context of data networks, such graphs would represent cases in which j could hear i, but i may not necessarily hear j. In this work, we shall be concerned with graphs whose links exist in both directions, called bidirectional graphs. Thus, we assume that "i hears j" is equivalent to "j hears i."

A mathematical description of a graph containing n nodes is obtained by constructing an n by n "connectivity matrix" whose $(ij)^{th}$ entry is unity (1) if i hears j and zero (0) otherwise. Such a topology matrix is shown for the network of figure 1a in figure 2b.

Graph vertex (or node) coloring theory attempts to find the minimum number of identifiers, which may be thought of as colors or letters of the alphabet, required to ensure that no two adjacent or "neighbor" nodes of a graph have the same identifier or color. In applying vertex coloring theory to the problem of finding the minimum number of TDMA-type transmission slots required for a given network (optimal "time-slotting"), we note it is necessary to assure not only that no neighbor j of a given node i shares a slot with i but also that no neighbor k of a neighbor j of i shares a transmission slot with i. Were this not done, some neighbor j of i could receive messages from two nodes at the same time (one from i and the other from one of its neighbors k



(a)



(c)

1	111111100000000000000000
2	111000111100000000000000
3	111100000111000000000000
4	101110000001110000000000
5	100111000000011100000000
6	100011100000000111000000
7	110001110000000001100000
8	010000111000000000111000
9	010000011100000000001110
10	011000001110000000000011
11	001000000111000000000001
12	001100000011100000000000
13	000100000001110000000000
14	000110000000111000000000
15	000010000000011100000000
16	000011000000001110000000
17	000001000000000111000000
18	000001100000000011100000
19	000000110000000001110000
20	000000010000000000111000
21	000000011000000000011100
22	000000001000000000001110
23	000000001100000000000111
24	000000000110000000000011

(b)

1	111111111111111111100000
2	111111111111000001111111
3	111111111111110000000011
4	111111100111111100000000
5	111111100001111111000000
6	111111110000011111100000
7	111111111100000111111000
8	111001111100000001111110
9	111000111110000000111111
10	111100111111000000001111
11	111100001111100000000011
12	111110000111110000000001
13	101110000011111000000000
14	101111000001111110000000
15	100111000000111110000000
16	100111100000011111000000
17	100011100000001111100000
18	110011110000000111110000
19	110001111000000011111000
20	010000111000000001111100
21	010000111100000000111110
22	010000011100000000011111
23	011000011110000000001111
24	011000001111000000000111

(d)

Figure 2. (a) 24-node network from figure 1a. (b) Associated topology matrix. (c) Network derived from (a) for purpose of optimal time-slotting. Nodes within two hops of one another are shown as being connected, and may not share slots. (d) Associated topology matrix for (c).

sharing the slot with *i*), resulting in a corrupted reception. (Such a time-slotting, which causes collisions between two transmissions, may be called a "false time-slotting" or a "miscoloring.") In other words, nodes reachable from a given node by two-hop paths, or nodes lying inside a "two-hop sphere" of a given node as illustrated in figure 3, are disqualified from sharing a slot with the given node. Consequently, the relevant vertex coloring problem must be applied not to the actual tactical data network graph, such as in figure 2a, but to a graph obtained by introducing into figure 2a links between all nodes *i* and *j* reachable from one another through one intermediate or relay node, i.e., by a path of length two. Application of this reasoning to the network of figure 2a results in the network of figure 2c with its associated topology matrix shown below the derived network, figure 2d. This derived network can be called the "square" of the actual network, as it can be shown that by squaring the original topology matrix one obtains a matrix whose nonzero elements *ij* correspond to those nodes *i* and *j* which are within two hops of each other, i.e., to those nodes that can be reached from each other along at least one path of length two (Christofides, 1975).

Although at first glance figure 2c appears to imply a complicated coloring or slot assignment algorithm, a simple lower bound on the number of time slots required in a tactical network can be obtained by noting that all the neighbors of a node *i* together with *i* itself form a set of nodes any of which is reachable from any other through, at most, one intermediate node and, therefore, that no member of this set may share a slot with any other. (It is a useful practice to consider node i to be a neighbor of itself and we shall do so in this work, unless specifically noted otherwise.) For example, consider as shown in figure 4 the neighbor set of node 2 in figure 2a: {1,2,3,4,5,6,7}. This set forms a set of nodes each of which is reachable from any other using a two-hop path. Consequently, they are neighbors of each other in the squared graph (figure 2c); hence, no two of them may share a time slot. It follows that the network of figure 2a cannot be assigned time slots (time-slotted) using less than seven slots—a number equal to the cardinality of the neighbor set of node 2 or, equivalently, to the sum of the members of row (or column) 2 in the topology matrix of figure 2b. Thus, by finding the most connected node or nodes *i* in the network (the node or nodes having the most neighbors) and adding up the elements in row (or column) *i* of the topology matrix, one obtains a lower bound on the number of time slots required by the network in an optimized time-slotting. We define the "degree of a network" as the highest row or column sum in the network topology matrix, which of course equals the cardinality of the neighbor set of the most connected, or highest degree node(s) (Christofides, 1975). Then we may summarize the above results as the Optimal Time-Slotting Theorem.

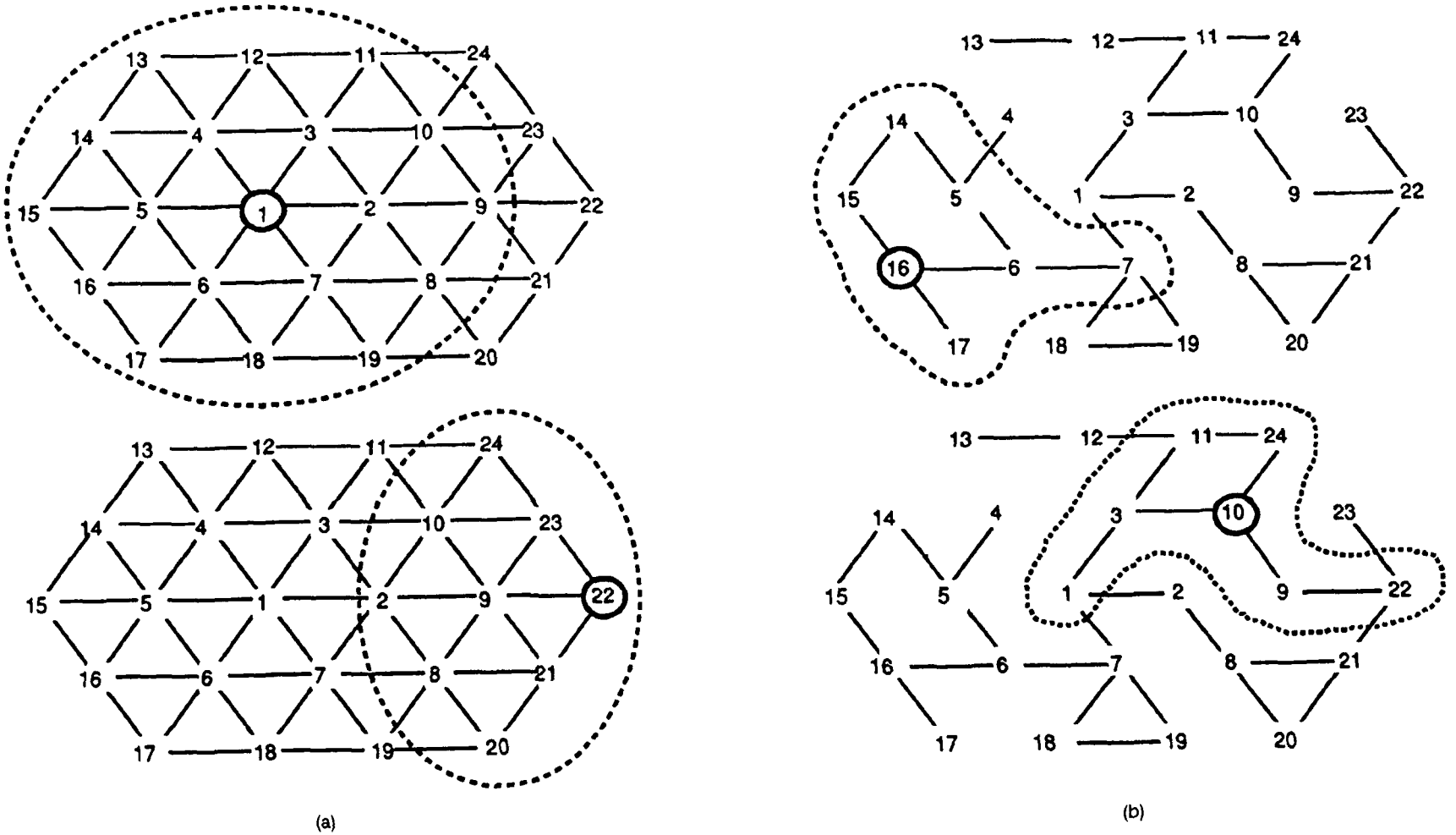


Figure 3. Illustrating two-hop "sphere" concept, useful for optimizing network time-slotting. Nodes within dashed "sphere" may not share a slot with circled node without causing corrupted receptions. (a) Two-hop "spheres" around nodes 1 and 22 in topology of figure 1a. (b) Two-hop "spheres" around nodes 16 and 10 in topology obtained by randomly changing existing links in (a) from connected to disconnected and back..

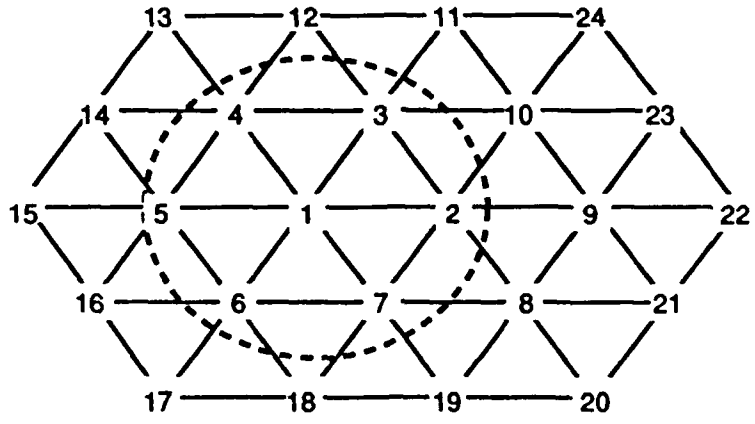


Figure 4. Basis of optimal time-slotting theorem. Nodes within dashed curve may not share time slots. Hence, for above network from figure 1a, optimal number of time slots, s, is equal to or greater than the cardinality, c, of the neighbor set of the maximally connected node(s), such as node 1

$$s \geq c \ .$$

In this case $c = 7$, so that

$$s \geq 7 \ .$$

Optimal Time-Slotting Theorem

The minimum number of time slots, s, required by a packet radio network for collision-free transmissions using an assigned slot channel access protocol or time-slotting is bounded below by the degree, c, of the network:

$$c \leq s \ . \tag{5a}$$

Using equation 3, we now have the following chain of inequalities for c

$$3 \leq c \leq s \leq n \tag{5b}$$

(ignoring as before the trivial case of networks of degree 2).

The Optimal Time-Slotting Theorem points the way to an efficient algorithm for assigning shared time slots for a slotted protocol. The algorithm makes use of the concept of the two-hop "slot sphere" around each node (see figure 3). The slotting algorithm begins by sorting the nodes in order of decreasing degree. In case more than one node has the same degree, the nodes having the most connected neighbors (most neighbors with maximal degree) are sorted higher than others. Then, slot A is assigned

to the most connected node in the network, and each of its neighbors is assigned distinct slots B, C, D, and so on. The algorithm then proceeds to the second most connected and unslotted node and assigns to it the first slot from the ones already used such that this node's two-hop slot sphere does not include any node to which that slot was already assigned. If no such slot exists within the slots used already, the next unused slot is assigned to this node. That node's neighbors are then assigned slots using the same criterion. The algorithm continues down the list of sorted nodes until all nodes have been time-slotted.

In practice, a slightly different form of the above algorithm is used, based on an algorithm described by Christofides (1975). The algorithm assigns slots to the highest degree nodes first, excluding nodes that lie within the two-hop slot sphere of a slotted node from being assigned the same slot. The time-slotting algorithm begins by sorting the nodes in order of decreasing degree, as in the previous algorithm. The node at the top of the list is then assigned slot A. A set called the "slot sphere of A," initially empty, is then augmented to include all neighbors of this node and all of their neighbors—nodes that cannot receive slot A. The next node on the sorted node list, which is not in the slot sphere of A set, is then assigned slot A, and the "slot sphere of A" set is augmented to include neighbors and neighbor's neighbors of this node. Continuing in this manner, the end of the node list is eventually reached, or else every node in the network is time-slotted. The algorithm then returns to the top of the sorted node list and finds the first node that has not been assigned a slot, whereupon it assigns slot B to this node. A set called "slot sphere of B," initially empty, is then augmented to include this node's neighbors and neighbor's neighbors. The algorithm is continued until each node has been assigned a time slot. A "pseudo-code" version of this algorithm is given below:

```
Define allowed range of slots (could be as large as desired, but
usually restricted to first few letters of alphabet);

Initialize each "SLOT_SPHERE_SET" to empty set;

Sort nodes of network in order of decreasing degree—
   break ties by putting nodes having the most neighbors with
   the highest degree above others.

CURRENT_SLOT  := A;

for i ranging from 1 to n loop

   if SLOT_SPHERE_SET of CURRENT_SLOT = entire network
   or if end of list reached then

        Return to top of list (i = 1)
        and begin loop with CURRENT_SLOT =
        slot_succeeding(CURRENT_SLOT);
```

end if;

If i an unslotted node and
if i not a member of of SLOT_SPHERE_SET of CURRENT_SLOT then

1. Assign CURRENT_SLOT to i

2. Add neighbors and neighbor's neighbors of i to "slot sphere of CURRENT_SLOT" set

end if;

end loop;

Done when all nodes are time-slotted.

Starting with one of the maximally connected nodes of figure 2a, say node 1, this algorithm time-slots the entire network using the eight slots {A, B, C, D, E, F, G, H} as shown in figure 5. In this case, and in some other cases, the number of slots required for time-slotting is one larger than the minimum given by the Optimal Time Slotting Theorem—the minimum is not achievable using this algorithm (or by any other means known to the author). However, experience obtained in the simulations discussed herein has shown that in most cases the minimum number of slots required is equal to, or only one greater than, the minimum given by equation 5.

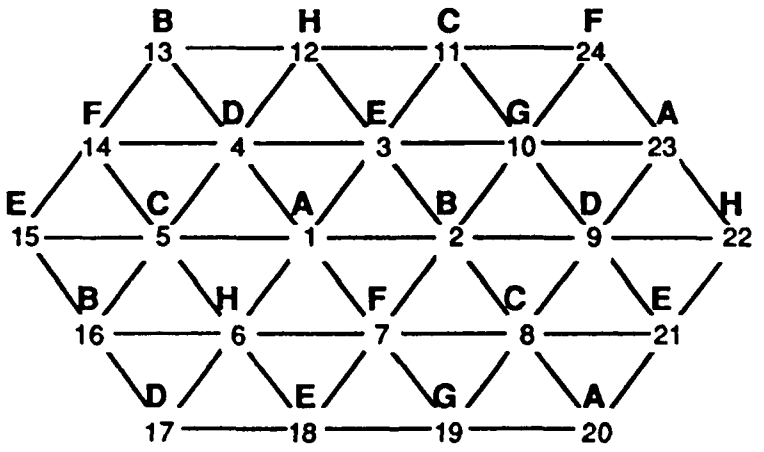


Figure 5. Time-slotting of network of figure 1a using two-hop sphere concept, or equivalently, Optimal Time-Slotting Theorem. In this case, per equation 4b,

$$F_{vc} = 1/s = 1/8 = 0.125 .$$

The algorithm can be applied to any subgraph of the network (i.e, to any part of the network containing some of its nodes and all of the links between these nodes). This fact is useful in developing distributed slotting algorithms, wherein each node assigns time-slots (or simply "time-slots") to its neighbor nodes, based on time-slottings assigned previously by other nodes to their neighbors, and transmits this information to its neighbors, who in turn perform the same algorithm. Such algorithms are discussed in the following section.

3.0 DESCRIPTION OF VOICE CONFERENCING ALGORITHMS DEVELOPED

We recall for convenience the description of a voice conference in a UHF-LOS network given in the first section of this work: a high-priority, accelerated data rate transmission of digitized and packetized voice in a UHF-LOS network between a caller node and several callee nodes, generally involving the use of relays, with all neighboring nonrelay nodes remaining silent (passive) while the conference is in progress. The algorithms for accelerating network data rates for voice traffic developed in the course of this work fall into two categories. The first is the caller-controlled routing/time-slotting (CRS) set of algorithms, and the second is the distributed routing/time-slotting set of algorithms. Under the CRS algorithms, the voice conference caller node decides, based on its own version of the global topology, the subgraph of the best relays for reaching the set of destination nodes of the voice call and the time slots to be assigned to these relays by applying the time-slotting algorithm specified in the previous section to the relay subgraph. Since the topology is dynamic, the instigator's topology information may be outdated and erroneous, and such an algorithm is susceptible, under fairly moderate levels of network dynamics (to be specified later), to generation of false circuits, which cannot be completed in the actual topology, or to corrupted transmissions due to false time-slotting.

Alternatives to caller-controlled algorithms are known as distributed routing/time-slotting algorithms. These algorithms give to each node only the power to decide which of its neighbors are to be involved in the voice circuit and which are not, assigning slots only to those neighbors it chooses to make part of the circuit. A node makes no such decisions regarding nodes that are not its neighbors, the motivation being that a node's topology information is most current for its neighbors, becoming less and less current for more distant nodes. Several types of distributed algorithms are investigated. One of these, called distributed routing/time-slotting (DRS), directs the caller to apply the routing algorithm developed in the case of the CRS algorithm but to restrict its choice of relays and time-slotting to "next relays" only, i.e., neighbor nodes its algorithm chooses as being in the relay subgraph. The caller transmits the information of chosen relays and time-slots to its neighbors in a channel acquisition packet (CAP)

(Merk, et al., 1988). Upon receipt of this packet, these chosen relays in turn execute the same routing/time-slotting algorithm, using their own versions of the topology, and choose next relays from among their neighbors not previously chosen. This process continues until all of the destinations are reached. These algorithms still assume some knowledge of topology more distant than neighbors and, hence, are subject, to almost the same extent, to the problems faced by the caller-controlled algorithms.

In view of the problems involved in trying to select a set of relays to reach a given callee set and to build as many redundant paths into the voice circuit as possible, a class of distributed algorithms, which incorporate the entire network into the voice conference, making every node a relay node, is investigated. These algorithms may be considered to be forms of network "flooding" algorithms, in that every node is asked to forward the same message traffic. In this sense, they represent a departure from reslotting aimed at transmissions between a caller and a few callees; instead they reslot the entire network, involving every node in reslotted voice transmissions. In this form of distributed algorithm, almost no knowledge of topology more distant than one's neighbors is assumed, but instead all of a given node's neighbors are incorporated as relays in the voice call, each of them being assigned a time slot.

Under this algorithm, known as distributed entire network time-slotting and reslotting (DENSR), the only assumption regarding the global topology is one made by the caller in setting the maximum number of slots in the time-slot cycle, which is done using the Optimal Time-Slotting Theorem and the caller's version of the network topology. The caller's algorithm, with the exception of the choice of number of slots in the cycle, is in turn repeated by each of the neighbors. The caller begins by assigning time-slots to itself and its neighbors using the slotting algorithm of the previous section and transmitting this information to its neighbors in a CAP. The neighbors upon receiving this packet become cognizant of the slot assignments of the caller and its neighbors, and assign time slots to each of their unslotted neighbors consistent with the slotting transmitted by the caller. This information is added to that sent by the caller and then retransmitted to the neighbors of the caller's neighbors, and so on.

The process can be viewed as the growth of a number of "slotting subgraphs" whose intersection includes the caller node and its neighbors. An example of such a slotting subgraph is given in figure 6b in which arrows are used to indicate the progress of the CAP and, hence, the growth of the slotting subgraphs. For example, node 14 has knowledge of the slottings of all nodes that occur on the path traced by the arrows from the caller node (24) to itself (node 14), as well as those slottings assigned by a node on this path to nodes just one hop off of this path (as indicated by an arrow pointing to such nodes from a node on the path). Thus, node 14 is cognizant of the slottings given to nodes {4,5,6,16,7,18,19,1,2,3,9,10,11,24} but not, for example, of the slotting given to node 15. The slotting subgraphs are not disjoint (a node can belong to

more than one slotting subgraph, depending on how the branching was carried out, see node 9 in figure 6, which also belongs to a slotting subgraph branching out to node 22), but each slotting subgraph grows in a self-consistently slotted manner, nodes on a subgraph being slotted in their order of growth from the caller node. Every node on a slotting subgraph has knowledge of slot assignments made on that subgraph prior to or coincident with its own slot assignment, but has no knowledge of slot assignments made on other slotting subgraphs, nor of slot assignments of nodes which grow on the same slotting subgraph after itself and its neighbors. Ideally, the union of all the slotting subgraphs is a consistent reslotting of the whole network. However, it is possible for nodes to get conflicting slot assignments. This happens when two nodes on separate slotting subgraphs make slotting assignments inconsistent with each other because each is unable to know what assignments the other has made. Examples of such misslottings are shown in figure 6: node 2 and 21 are assigned conflicting slots as are nodes 14 and 15, since members of these pairs are on separate slotting subgraphs. Certain provisions to allow resolution of such slotting conflicts by individual nodes "reslotting" themselves have been built into this algorithm, as described in a later subsection.

Because of the incorporation of the whole network into the voice conference time-slotting, the DENSR algorithm forces the use of more slots s per slot cycle than generally are employed in the CRS and DRS algorithms, resulting in a lower channel utilization fraction F_{vc} (equation 4b). It therefore represents a departure from the goal of employing the shortest time-slot cycle possible. But the advantage gained thereby is that the reslotting and, hence, the integrity of the voice traffic is more capable of withstanding network topology changes than either the CRS or DRS algorithms because, first, it makes almost no assumptions about connectivity beyond one-hop distances, and, second, redundant pathways often available in the network are retained by this algorithm for use in the voice transmissions, so that in case one of them breaks down, another might be available to relay the message to the callee. An example of this redundancy is afforded by figure 6, which shows two possible pathways from node 24 to node 8: one involving the chain 24-10-3-1-2-8 and another involving the chain 24-10-9-22-21-8.

An embellishment of the DENSR algorithm, called distributed entire network time-slotting and reslotting with topology updating (DENSRTU), allows, after the time slotting described above, for each node in the network to send back to the caller its most current local topology information (which nodes are neighbors to a given node). Using the aggregate of this updated information, the caller executes a caller-controlled routing/time-slotting (CRS) with the result that the final routing/time-slotting for the circuit can be done with greater reliability and minimum use of relays and time slots.

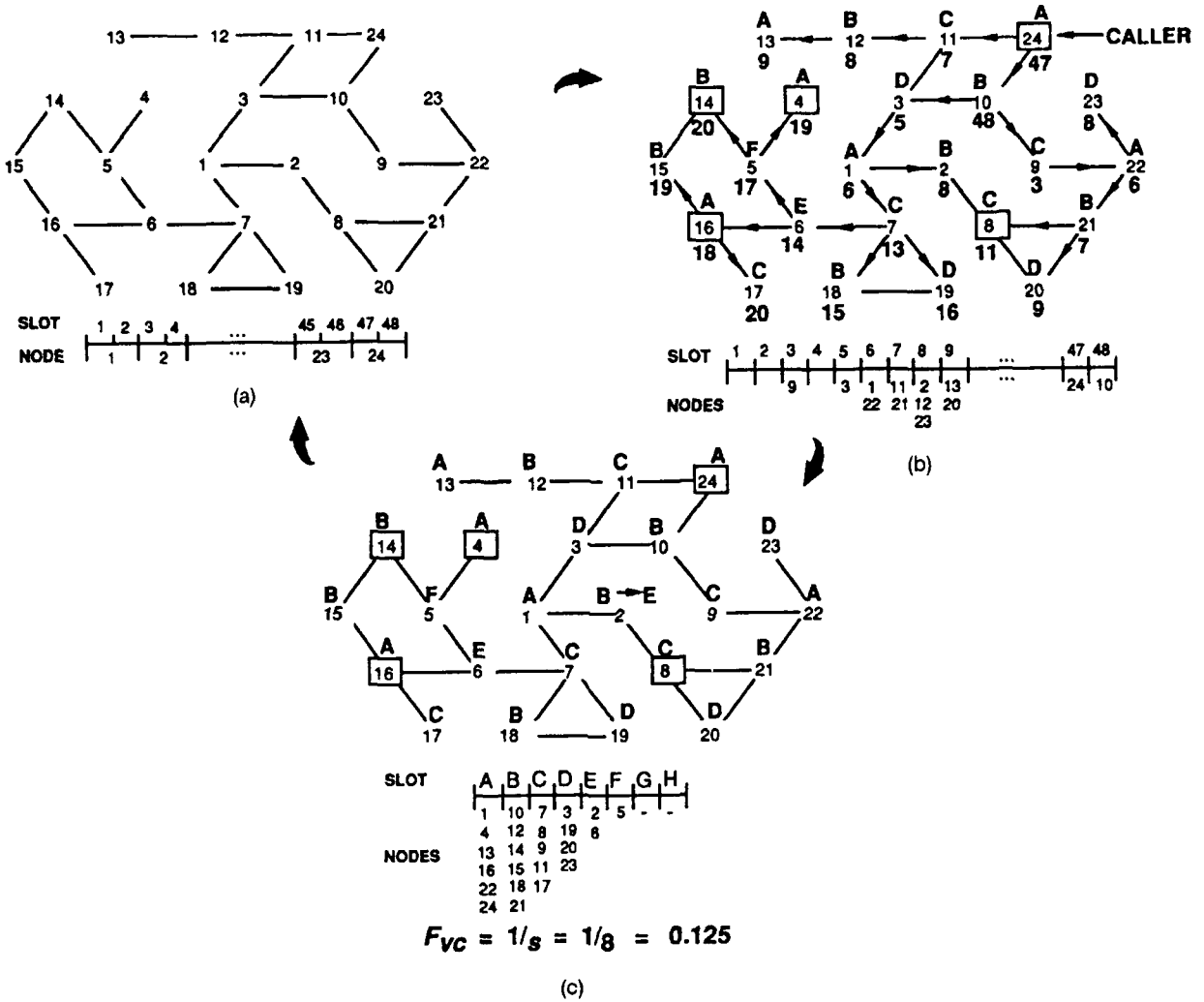


Figure 6. Evolution of network under Distributed Entire Network Time-Slotting/Reslotting (DENSR) algorithm. Squares show conference members. (a) Network under HAMA. (b) Temporary (CAP) slot assignments (lower bold numbers) and time-slotting (upper bold letters) initiated by node 24 (arrows show progress of CAP). (c) Network in final time-slotting (node 8 reslots from B to E, but 14-15 slotting conflict remains unresolved.) Network reverts to HAMA at termination of conference.

Since all the algorithms require the network to send high-priority routing/slotting messages in a CAP (Merk, et al., 1988) using chosen relays along paths to destinations that may be several hops away, it was thought inefficient to rely on the usual TDMA protocol for transmission of this packet. Considering that for each hop roughly one-half of a net cycle would be required under TDMA for the relay's slot to arise for the CAP transmission, and that the voice transmission itself may be no more than a few seconds long (Merk, et al., 1988), or possibly only a few net cycles long depending on the TDMA slot size, the overhead time or bandwidth required to set up the circuit under the TDMA or HAMA protocol could be longer than the actual voice traffic time. Since voice in a tactical network is the highest priority traffic, methods of interim "one-way" time-slotting to expedite the CAPs along their journeys to the callees are devised. Note that this is a "one-time" time-slotting used only for the single transmission of the CAP, after which all transmissions in the voice conference are done using the voice-slotting discussed above. A method for such an interim time-slotting outlined by Merk et al., (1988) is suitable for caller-controlled point-to-point algorithms: each relay may use the next available slot belonging to a relay, or neighbor thereof, which has already received the CAP. This algorithm is easily modified to apply to the caller-controlled routing/time-slotting algorithm for voice conferencing: the caller can simply sort in order of occurrence the TDMA (or HAMA) slots of its chosen relays and neighbors of these relays (available slots) and reassign them to the relays in the manner most efficient for progress of the CAP to its destinations. In running distributed algorithms, however, nodes make slotting decisions on disjoint trees as discussed above, so it is not possible to assign without conflict the slots of previous nodes in the history of the CAP's progress. Hence, a different interim slot assignment algorithm is used. The basis of this algorithm is similar to the time-slotting algorithm discussed in section 2: a chosen relay may use any slot belonging to a node outside of a certain "slot sphere" centered on itself. Because of the possibility of slots being handed off under the HAMA protocol, it is necessary to ensure that a relay node be assigned the next available slot belonging to a node more than three hops away (rather than just two hops away) from itself to transmit its CAP. As will be discussed in a later section, this CAP slot assignment algorithm is found to result in relatively quick reslotting and channel acquisition, although not without causing occasional slotting conflicts fatal to a certain percentage of voice conferences in the simulations.

We provide below further details of the algorithms devised. In the discussion that follows, "identifier," when applied to a given node, is the integer corresponding to the node's regular TDMA time slot, per figure 1b, and "time slot" and "voice slot" are both used to refer to the node's assigned slot during the voice conference, which for current purposes can be thought of as one of the first 8 or 10 letters of the alphabet. The term CAP-slot or CAP-slotting will be used to refer to the one-time slotting assigned to expedite the CAPs to the callees.

3.1 CALLER-CONTROLLED ROUTING/TIME-SLOTTING (CRS) ALGORITHM

In this algorithm, the caller is assigned the task of choosing and time-slotting all relays to the destinations. The routing algorithm is based on Djikstra's shortest path algorithm, discussed by Christofides (1975). Using Djikstra's algorithm and the caller node's own version of the network topology, the caller establishes a shortest path to each of the callees. The union of these paths gives a possible subgraph to be used for the call. A subgraph with fewer relays is often obtainable by first constructing the shortest path to the nearest of the callees (callee fewest hops away) and then tracing back from the next nearest callee the shortest path to the caller until this path meets the one from the previous callee. Continuing in this manner until all the callees have shortest paths constructed to the preexisting subgraph involving callees closer to the caller than the current one, results in a subgraph involving close to the minimal number of nodes for the given topology. This subgraph is then voice time-slotted using the slotting algorithm of the previous section. Identifications of chosen relays and their voice data time slots are written into the CAP, along with the CAP-slotting discussed above for the caller-controlled algorithm. The CAP is then broadcast by the caller on its normal TDMA or HAMA slot. The caller waits to hear the retransmission of the CAP by the chosen relay(s) on their interim CAP slots and then begins to transmit voice packets, which initially include only its node identifier, on its own voice time slot. After transmission of the CAP and verification of retransmission by the next neighbor relays on the voice conference subgraph, each node begins to transmit acknowledgment of receipt of the CAP using the voice conference time-slotting assigned to it by the caller. All voice packets initially include only the transmitting node's identification, but as packets are received from neighbors, each node adds (unions, or "ORs in") the identification of other nodes that acknowledge receipt of the CAP. In this way, if the channel acquisition has been successful, the caller eventually receives acknowledgment of CAP receipt from each designated member of the circuit. At this point the caller commences voice data transmission. Nodes not on the time-slotted relay list remain silent or "passive" during the call (abort normal TDMA or HAMA transmissions) until they detect that voice transmissions have ceased. They do this by monitoring transmissions of their neighbors and determining when the latter have ceased voice transmissions (ceased to jam the passive node receptions) for at least a time period equal to twice the number of nodes in the network multiplied by the number of voice slots per cycle multiplied by the length of a voice slot.

If the channel is not successfully acquired within a reasonable amount of time (taken to equal 3.5 net cycles) the caller reverts to TDMA or HAMA data mode. This is also true of other nodes converted to voice conference protocols. Voice packets have a sequence field in them, and they are assigned a cyclic sequence number between 0 and 16 at the time of their creation. If any node does not receive the correctly

numbered voice packet after five voice slot cycles, it issues a "route destruction packet" and the voice circuit is considered to have ended abnormally.

A pseudo-code summary of the CRS algorithm is the following:

```
for i in 1..NUMBER_OF_NODES_IN_NETWORK loop
   If VOICE_CONFERENCE_REQUEST in VOICE_CONFERENCE_QUEUE then
      1. i constructs shortest path to each callee
         using Djikstra's algorithm;

      2. i selects relays for reaching closest callee;

      3. i traces back path from second closest callee until it joins path
         from closest callee, third closest, etc., until all callees are
         connected into voice conference subgraph;

      4. i assigns voice slots based on Optimal Time-Slotting Theorem and
         associated algorithms;

      5. i assigns CAP slots using the next available slot
         belonging to voice conference subgraph members and neighbors,
         in a manner so as to expedite CAP progress;

      6. i transmits CAP on caller's first HAMA or TDMA slot;

   end if;
end loop;
   if CAP_RECEIVED then

      Each relay transmits CAP on its CAP slot;

      Each relay transmits union of all members of voice conference subgraph
      from whom acknowledgments have been received;

   end if;

      Caller begins voice after receiving acknowledgment from all
      members of subgraph;

      i registers normal termination (successful)
      after all voice transmissions complete.
      All nodes revert to TDMA or HAMA;

      Abnormal termination (unsuccessful) if some member(s) did not receive
      every voice packet. Route destruction packet (RDP) sent to caller.
      All nodes revert to TDMA or HAMA;
```

3.2 DISTRIBUTED ROUTING/TIME-SLOTTING (DRS) ALGORITHM

The main difference between this algorithm and the caller-controlled algorithm of the previous subsection is that each node uses its local version of the network topology

to deduce a near-optimal call-routing subgraph, but adds only the nodes belonging to the intersection of that subgraph with its neighbors set to the call routing subgraph received by it from the previous relay (neighbor that transmitted the CAP to the node in question). Thus, the caller node includes in the subgraph itself and those of its neighbors on its call routing subgraph. It time-slots these neighbor nodes only, using the slotting algorithm discussed in the previous section. In the case of the caller, this algorithm results in each relaying neighbor of the caller receiving its own time slot, since they are all within each other's two-hop sphere. The caller also assigns a CAP slot to each chosen neighbor, as described above, using the next available TDMA or HAMA slot from a node that lies outside the neighbor's three-hop sphere. Additionally, the caller determines the total number of time slots to be used in the voice conference time-slotting, which is assumed equal to the integer one larger than the degree of the network as observed in its local version of the network topology. The caller writes the identifiers of the chosen relay, their voice slot assignments, their CAP slot assignments, and the total number of voice slots into a CAP and transmits it on its own TDMA (or first HAMA) slot. Each chosen neighbor then performs the same algorithm as the caller, except for choosing the total number of time slots, using its version of the topology. In assigning time slots to their chosen relays, each node applies the algorithm of the previous section to the subgraph consisting of all the previously chosen relays whose identifiers are written into the CAP received by the node in question. As has been noted above in the discussion of the DENSR algorithm, the latter identifiers may not include all relays chosen by all members of the voice circuit, as nodes on a separate "slotting subgraph" from the node in question have no way of writing information into the given node's CAP. Thus, circuits with slotting conflicts may be created, this being one of the drawbacks of this algorithm. However, nodes on the same slotting subgraph are time-slotted in a self-consistent manner.

Nodes not chosen as relays but which nevertheless are connected to chosen relays (i.e., passive nodes) are under this algorithm required to broadcast a "going silent" message, so other nodes will not try to choose them as members of the call subgraph (as they might otherwise try to do since their version of the topology is different from that of any other node). Such passive nodes are also assigned a "one-time" slot to speed up this notification process, this information being written into the CAP. Some further refinements are introduced into the DRS algorithm to lower the likelihood that the conference subgraph deduced by each node is inconsistent with those already chosen. For example, since the CAP received by a node chosen to be on the call-routing subgraph carries a partial history of relays chosen up to the node executing the algorithm, this information is used to decrease the probability the node will choose a subgraph that includes nodes previously told to go silent.

After transmission of the CAP and verification that all neighbors have either retransmitted the CAP or broadcast a "going passive" notification, each node transmits, using the voice time-slotting, acknowledgment of CAP receipt as well as acknowledgments of receipt received from any other nodes, as in the CRS algorithm. The caller begins transmission of voice packets upon receiving acknowledgment from all the destinations. Passive nodes return to active after establishing that their receptions are uncorrupted for a time equal to the number of nodes in the network multiplied by the number of voice slots per cycle multiplied by the length of a voice slot. Protocols for reversion of all other nodes to TDMA or HAMA data modes are the same as that of the CRS algorithm.

A pseudo-code summary of the DRS algorithm is the following:

```
for i in 1..NUMBER_OF_NODES_IN_NETWORK loop

   If (VOICE_CONFERENCE_REQUEST in node's VOICE_CONFERENCE_QUEUE) or
   if (CAP from another node has been received) then

    1. i constructs shortest path to each callee
       using Djikstra's algorithm and node's local topology matrix;

    2. i selects the relays for reaching closest callee;

    3. i traces back path from second closest
       callee until it joins path from closest callee,
       from third closest callee, etc., until all callees are
       connected into voice conference subgraph;

    4. i assigns voice slots only to neighbors
       who are on voice conference subgraph,
       based on Optimal Time-Slotting Theorem and
       associated algorithms;

    5. i assigns CAP slots only to neighbors who are on
       voice conference subgraph, using the next available slot
       belonging to node outside of given neighbor's "three-hop sphere";

    6. If i is caller then
            transmits CAP on caller's first HAMA or TDMA slot,
       else
            transmits CAP on relay's assigned CAP slot;
       end if;

   end if;
end loop;

   Each relay transmits union of all members of voice conference subgraph
   from whom acknowledgments have been received;
```

i begins voice data packets after receiving acknowledgment from all members of subgraph;

i records normal termination (successful)
after all voice transmissions complete.
All nodes revert to TDMA or HAMA mode;

If some member(s) did not receive every voice packet,
route destruction packet (RDP) sent back to
caller. Abnormal termination recorded. All nodes revert
to TDMA or HAMA mode;

3.3 DISTRIBUTED ENTIRE NETWORK TIME-SLOTTING/RESLOTTING (DENSR) ALGORITHM

In the DENSR algorithm, each node assumes only knowledge of whom it can hear, except the caller chooses a cardinality of slots which it considers will be adequate for the entire network based on the degree of the network observed in its version of the global topology. Instead of trying to route voice packets on a specific circuit, the aim in this algorithm is to time-slot the entire network using the predetermined number of slots, thus avoiding choosing specific paths that may no longer exist and retaining the redundancy of paths often inherent in a network. Thus, the caller time-slots itself and all of its neighbors, as well as assigns them an interim time-slotting for broadcasting their CAPs according to the "three-hop sphere" procedure outlined at the beginning of this section. The caller writes this information into its CAP and broadcasts it on its regular TDMA time slot. After verifying that each of its neighbors has retransmitted the CAP, the caller begins broadcasting on the time slot assigned to itself for the voice circuit. Initially, it broadcasts only its node identifier. Each neighbor performs the same algorithm (except of course for setting the total number of slots) in a manner consistent with slottings assigned by the caller and, thus, continues the process of "slotting subgraph" growth initiated by the caller. The slotting subgraphs grow from their common roots consisting of caller node and neighbors of this node until all of the callees have been reached.

After transmission of the CAP, each node puts its identifier into each voice packet and cumulatively unions (adds in) identifiers of all nodes who acknowledge receipt of the CAP. The caller begins transmitting voice packets after receiving acknowledgment from all destinations, as in the distributed routing/time-slotting algorithm. In case of failure to receive the correct sequence of voice packets, a node remains passive for a certain time before resuming normal data transmissions, in case neighboring nodes do receive the correct packets. Passive nodes revert back to regular data transmissions if their receptions are uncorrupted for at least a time equal to the product of three, the number of nodes in the network, the number of voice slots per cycle, and the length of

a voice slot. All other criteria for returning to regular data mode are the same as for the CRS and DRS algorithms.

A pseudo-code summary of the DENSR algorithm is the following:

```
for i in 1..NUMBER_OF_NODES_IN_NETWORK loop

   If VOICE_CONFERENCE_REQUEST in node's VOICE_CONFERENCE_QUEUE or
   if CAP from another node has been received then

      1. i assigns a voice slot to each neighbor
         that does not have one assigned in the received CAP (if any),
         consistent with the slotting assignments
         found in its received CAP.

      2. i assigns a CAP slot to each neighbor
         that does not have one assigned in the received CAP (if any),
         consistent with the slotting assignments found in the CAP.

      3. If CALLER then i transmits CAP on
         first HAMA or TDMA slot,
         else on first assigned CAP slot;

   end if;
end loop;

   Each relay transmits union of all members of voice conference subgraph
   from whom acknowledgments have been received;

   Caller begins voice after receiving acknowledgment from all
   members of subgraph;

   Caller records normal termination (successful)
   after all voice transmissions complete.
   All nodes revert to TDMA or HAMA mode;

   Nodes not receiving correct sequence of packets remain silent for a time
   so that other nodes may transmit, in case they do receive the correct
   sequence;

   Abnormal termination (unsuccessful) if some callee(s) did not receive
   every voice packet (RDP sent to caller). All nodes revert
   to TDMA or HAMA mode;
```

3.4 DISTRIBUTED ENTIRE NETWORK TIME-SLOTTING AND RESLOTTING WITH TOPOLOGY UPDATING (DENSRTU) ALGORITHM

In this algorithm, which is an embellishment of the DENSR algorithm, each node includes in its acknowledgment packet its latest knowledge of which nodes are its neighbors. This information is accumulated (unioned, or "OR'd in") at each node.

Thus, when the caller has received acknowledgment from all the destinations, it also has built an updated partial network topology from which it may construct a caller-controlled final routing/time-slotting. The caller broadcasts this final information in a second CAP, which is relayed using the voice time-slotting set by the first CAP. After delaying a few time-slot cycles (to avoid possible collisions between different time-slottings), each node on the final circuit begins transmitting on the final time-slotting (caller transmits first voice packet) and nodes not on the circuit become silent.

Pseudo-code for this algorithm is the same as that for the DENSR algorithm, as far as the main loop is concerned, but is completed in the following manner:

1. All nodes transmit local neighbor's information along with their acknowledgement packets.

2. After receiving acknowledgments from all callees, caller constructs a caller routed/time-slotted (CRS) subgraph from topology information transmitted to him by all nodes. Transmits new slotting and routing information on current voice slot.

3. Network behaves as under CRS algorithm, omitting acknowledgments by relays.

3.5 ALGORITHM FOR LIMITED RESLOTTING

Voice and CAP slot conflicts can arise in DENSR and DENSRTU algorithms for the same reason they may do so in the distributed routing/coloring algorithm: a node may not have information regarding slots assigned to nodes in its vicinity, as these assignments may have been made by a node on a different slotting subgraph from its own, the CAP to those nodes having propagated along a different branch in the network. However, since each node has fairly current knowledge of its neighbor set, it is possible in some cases to resolve conflicting voice slot assignments, which may arise for reasons mentioned above. This is implemented in the DENSR and DENSRTU algorithms by having each node write into the CAP acknowledgment packet a list of all voice slots observed (by receipt of transmissions) to be in use by its neighbors and the identifier of any neighbor whose transmissions it cannot receive. If more than one such neighbor exists, that neighbor with the lowest number identifier is chosen for notification. Since each node transmits the slots in use by its neighbors, neighbor nodes receiving these transmissions are able to construct sets of slots in use by nodes within their two-hop sphere. If they are notified that their transmissions are not being received by one of their neighbors, they decide to change their slot from their assigned voice slot to another available slot, if any. This procedure, which may be called a limited form of reslotting, is found to improve the performance of the DENSR and DENSRTU

algorithms. An example of a successful reslotting under this algorithm is shown by node 2 in figure 6c, which was told that its transmissions were not being received by node 8; however, the false slotting between nodes 14 and 15 of this figure remains unresolved under this algorithm, since neither of the nodes could inform the other that their transmissions were not being received (a node is assumed not to be able to receive transmissions on the same slot that it broadcasts). Further discussion of the reslotting algorithm is deferred to the conclusion section.

Pseudo-code for this reslotting algorithm is as follows (assuming CAPs have been received and network is time-slotted):

```
for i a node in the network loop

    1. record and transmit identifiers of voice slots that are in use
       by neighbors of i (N1_COLORS);

    2. record (N2_COLORS) which voice slots are in use by neighbors of i and
       (neighbors of neighbors of i) from information received from item
       (1) transmissions (N1_COLORS);

    3. record which nodes known to be neighbors are not being received
       uncorruptedly;

    4. notify the node (CORRUPTED_NOTIFICATION) with lowest
       identifier out of the nodes recorded in (3);

    5. if (CORRUPTED_NOTIFICATION_RECEIVED) then

       change voice slot to a different available slot, if any, based on
       local version of N2_COLORS;

   end_if;

end loop;
```

A further problem sometimes was observed to arise when a node may not have a completely current list of its neighbors (locally erroneous topology), owing to very recent changes in the topology of which the node has not had time to be apprised by receipt of TDMA or HAMA transmissions. Hence, a node may receive a CAP but not be included in the voice and CAP slotting lists. In this case, the protocol directs the node to choose a voice and CAP slot consistent with the nodes that have so far grown onto its slotting subgraph. This procedure is not failsafe, but is observed to help certain circuits to succeed. The inverse problem, in which a node is perceived to be a neighbor but no longer is, also arises but does not appear to be as deleterious to the voice circuits studied.

The above algorithms were subjected to computer simulation in the process of developing and improving them to the point described herein. A summary of the

simulation program used for this development and input parameters to the simulation is the subject of the next section.

4.0 DETAILS OF SIMULATION OF ALGORITHMS

The voice conference algorithms were simulated using EVADA (Mumm and Ollerton, 1990), an Ada-based discrete event simulation program. This is an event-oriented simulation environment specially designed for simulation of network protocols. The use of the simulation system involves the following steps:

1. Reading of inputs and initializing of network variables.
2. Seeding the network "event queue" with events sequenced by execution time. These events may in turn schedule other events, or reschedule themselves.
3. Transfer of control to the event on the front of the event queue.
4. Execution of events in the order indicated by their schedule time, according to a simulation clock maintained during the entire simulation, and updated as events are removed from the ready queue.
5. Continuation of (4) until a "stop simulation" event rises to the front of the ready queue.

Examples of events are the transmission of Navy Tactical Data System (NTDS) messages (see Gutman, et al., 1988) for a description of NTDS messages), creation or breakage of a link between two nodes to simulate dynamics in the topology, generation of a request for a voice conference, and execution of voice-related protocols. For lack of space, it is not possible to include in this work many details of the simulation, a summary of which is found in Mumm and Ollerton (1990). We restrict ourselves to details relevant to the simulation of voice conferencing algorithms.

Knowledge of which nodes are one's neighbors (local topology information) is obtained through the agency of a small "slot-owner subslot," which precedes the normal TDMA or HAMA data slots and is one-hundredth the size of one of the latter. Since this slot is always used by its owner in the TDMA or HAMA scheme, failure to hear the slot at its scheduled time is conclusive evidence that the owner node is no longer a neighbor and thus is to be "zeroed" in the local topology table. Conversely, if a new link should appear, an additional node's slot-owner subslot will be heard, and the node's entry in the appropriate row of the topology table will be changed to "unity" within a time period of one TDMA or HAMA net cycle after the establishment of the link.

As implied above, there is for these simulations a constant background traffic of NTDS packets, which are "broadcast" packets meant to be routed to every node in the

network. These packets are simulated as empty or "dummy" packets in the simulation, except for certain relevant information contained in each of them. This information is the source node's identifier and a list of identifiers of all nodes known by it to be its neighbors at the time of generation of the packet—the "source node neighbors" field. Each node, upon originating a broadcast packet, records in this field all the nodes it has been able to hear during the last TDMA or HAMA epoch. When a node receives a NTDS packet, it inserts this information into the row corresponding to the source in a locally maintained global topology matrix. This is the only information a node has of topology in regions of the network more distant than one hop from itself (since UHF-LOS is assumed to be the only communication medium available to the nodes) and it is this matrix that contains the topology used in all routing algorithms, i.e., the CRS and DRS algorithms. Since topology can change randomly with time (with certain modeling constraints in this simulation as discussed below), it is evident that the success of voice conference routing algorithms depends critically on the currentness of the information in the global topology matrix, which is in turn dependent on the particular store-and-forward protocol in place in the simulation, and also on the rate of generation per node of broadcast messages. The latter is a parameter that can be specified in the packet generation event by the user of the simulation. If this rate is very low, broadcast packets will be generated infrequently and the global topology will be seldom updated. If this rate is high, a similar effect can ensue, as each node's message buffer queue becomes saturated by its own generated packets preventing significant relaying of packets from other nodes. In this case most of the information obtained by any node is only local in nature. Two different packet generation rates per node were simulated in the course of this study—one resulting in a packet being generated every net cycle by each node, another resulting in a packet being generated every five net cycles by each node.

For all the voice conference algorithms, it is necessary to model topology dynamics. Two models of topology dynamics are used in these simulations: one in which initially existing bidirectional links are randomly changed from connected to disconnected and back (hereafter called Model 1), and another in which motion of one or more nodes relative to the rest of the network is modeled (hereafter called Model 2), in the course of which some existing UHF-LOS links are destroyed and new ones created.

The random destruction and re-establishment of existing links (Model 1) is chosen as a model of link dynamics easily quantifiable in terms of one parameter and also restricting the degree of the network to be no larger than an initially prescribed value. The latter condition is adopted because it induces a milder set of topology changes than those wherein the degree of a node (cardinality of its neighbor set, including itself) could become as large as the number of nodes in the network, and thereby permits running the voice algorithms with a preset number of voice slots.

On this model of network dynamics, the network is initialized as a 24-node network with the topology of figure 2a-b. The links in this topology are considered to constitute the totality of possible links. All link changes, thus, consist of randomly "flipping" one of these links—if the link is "up" (nodes connected) it is randomly set to "down" (nodes disconnected) and if down it is randomly set to up. An additional constraint is that the network is not allowed to become disconnected. This simulation of link changes corresponds to nodes in a network maintaining approximately constant relative positions with link changes resulting from a random excursion in the x-y plane of each node about a fixed position (similar to Brownian motion), or to the random appearance and disappearance of link-destroying jammers. Note this type of topology change precludes links appearing between nodes in initially distant parts of the network. Thus, the topology changes studied in these simulations represent a small, but important, class of possible link changes. Since the degree of any node is restricted to a maximum value (7 according to figure 2) the number of voice slots per cycle of DRS and DENSR algorithms, as well as the initial number of voice slots per cycle of the DENSRTU algorithm, is preset to a value of 8—one larger than the degree of the network. This was found to be an acceptable value for this parameter and avoided the necessity of the caller having to estimate the required number of slots. This represents a slight departure from the description of the algorithms in the previous section, but was adopted to simplify the simulation.

An important parameter for the simulation of Model 1 of network dynamics is the mean number of link changes per net cycle, λ_{link}. This parameter is used as the mean of an exponential process simulating the interarrival rate of link changes. Because of the initial configuration and the link change algorithm just described, it is evident that early in the simulation link flips will be almost all from "connected" to "disconnected." To overcome this initial bias, 100 link flips were performed before any actual message traffic was simulated.

Link dynamics Model 2 simulates the translational motion in the x-y plane of one or more nodes relative to the remainder of the network initially having the topology of figure 2a-b and the attendant destruction and creation of links due to these spatial motions. As nodes move relative to the others, they may exceed the UHF-LOS (15 nmi) between themselves and their initial UHF-LOS neighbors and may enter the UHF-LOS of other nodes. Thus, all original links between such nodes and their neighbors may be broken and new ones established. This may give rise to topologies in which nodes have significantly higher degree than in the initial topology of figure 2a-b. This second model, thus, simulates another significant class of topology changes.

The network for Model 2 is initialized with x-y coordinates assigned to each node, the topology being determined by whether nodes are within each other's UHF-LOS based on the two-dimensional distance formula. Two nodes of the network are then

given an initial velocity, and their positions are checked every few net cycles to determine which links have been broken and which have been created.

During most of the simulation time, the network sends broadcast packets. The generation of voice conferences is modeled by creation of a voice conference request in a "voice conference queue" maintained at each node. The interarrival time of voice conference requests per net cycle is modeled as an exponential process with corresponding mean of λ_{voice}. A voice conference is instigated by a node finding a conference request in its voice conference queue. This node then becomes the caller. The voice conference request contains the identifiers of four randomly chosen callees, this cardinality being held constant in these simulations. Also, a constant value of

$$\lambda_{voice} = 20.0 \tag{6}$$

is used for all simulations.

Voice conference traffic is the highest priority traffic in the simulation and preempts all broadcast traffic. From this it can be seen that the currentness of network topology information at each node suffers when a voice conference is in progress, as no regular data packets having the source node's topology information as one of their fields are propagated by nodes involved in or silenced by voice traffic. Each voice conference in the simulation, thus, leaves in its wake a "vacuum" of topology information at the participant nodes. Moreover, since the interarrival rate of circuits is an exponential process, occasionally a circuit in the simulation may be generated while another one is still in progress. In this case the former circuit is almost sure to fail, so even a perfect routing/time-slotting algorithm will not guarantee success in all cases. In the context of the simulation, this means that the theoretical limit to which performance of the algorithms should be compared is something less than 100%. Further discussion of this point is given in the next section.

The duration of a voice slot is another parameter that can be determined by the user at the outset of the simulation. TDMA or HAMA data slots were assumed to be 1 second in duration and to carry 9600 bits of information. Voice slots in principle could be as long as data slots, but in these simulations were assumed to be smaller. A ratio of voice slot size to data slot size of 0.0691 was used for all the algorithms except the DENSRTU algorithm. In the latter algorithm, in view of the large amount of information required by nodes to send their local topology information back to the caller, the larger ratio of 0.35 was used. After initiation of voice data transmissions by the caller, the voice conference simulation itself consists of the caller sending enough packets for the given voice slot size to transmit about 50 kilobits of voice, followed by a similar length reply from a randomly chosen callee out of the four, and completed

by a second similar length transmission by the caller. At a data rate of 9600 bps, this corresponds to 77 packets of voice sent in the simulation by each speaker using a slot ratio of 0.0691 and 15 packets using a slot ratio of 0.35, numbers that are used in the simulations.

All simulations were done using the two-slotted HAMA channel access protocol for NTDS transmissions outlined in Norvell, et al., (1985). This protocol was chosen over TDMA as it is of local interest to Naval Ocean Systems Center. Thus, one net cycle in this simulation corresponds to $2 \bullet 24 = 48$ slots.

5.0 RESULTS OF SIMULATION AND DISCUSSION

5.1 DISCUSSION OF SIMULATION OF TOPOLOGY DYNAMICS MODEL 1

In studying the performance of the voice conference algorithms of section 3.0 under the topology dynamics of Model 1, simulation of the algorithms was performed at values of $\lambda_{link} = 0.1$ up to $\lambda_{link} = 40.0$. The results of this parametric study are shown in figure 7 for the case in which the NTDS packet generation rate at each node is unity per net cycle, and in figure 8 for the case in which this rate at each node is unity per every 5 net cycles. Further information on each algorithm is illustrated in figures 9 and 10, which show column graphs with error bars representing the mean and standard deviation of acquisition time and of duration of voice conferences in terms of regular data slots (from transmission of CAP by caller to transmission of final voice packet by caller after speaker change), mean and standard deviation of the number of voice slots per cycle under each algorithm, and statistics on mean and standard deviation of number of conference members per conference. Recall that the distributed routed/slotted algorithm and the DENSR algorithm both used a preset number of eight voice slots per cycle under topology dynamics Model 1.

As noted above, because voice conference requests are generated in these simulations with an exponential interarrival rate, some percentage of conferences are expected to start or be requested while a previous one is in session or is acquiring the channel. Such conferences are almost certain to fail because many nodes of the network are already participants of a previous conference. Thus, the theoretical maximum success rate for voice conferences (ordinates of figures 7 and 8) is a percentage less than 100. An estimate of the maximum expected success rate in the limit of perfect knowledge of network topology at each node can be obtained by noting that for exponentially distributed interarrival rates the probability P_{nocoll} that no voice conference request will arrive within a time T of the last previous request is given by

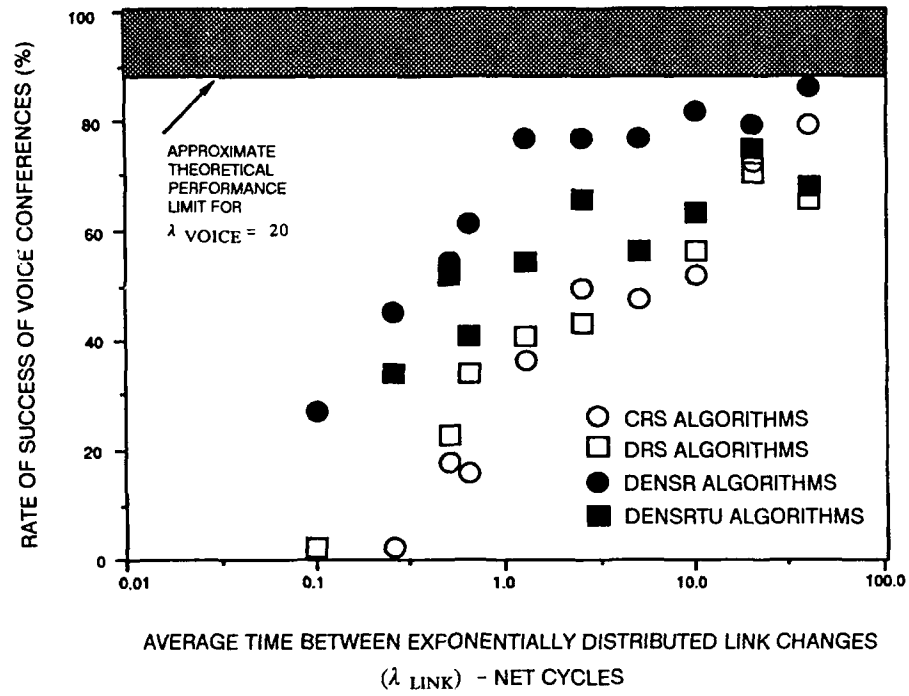


Figure 7. Graph of voice conferencing simulation results: 24-node network (figure 2a), one NTDS packet per node per net cycle.

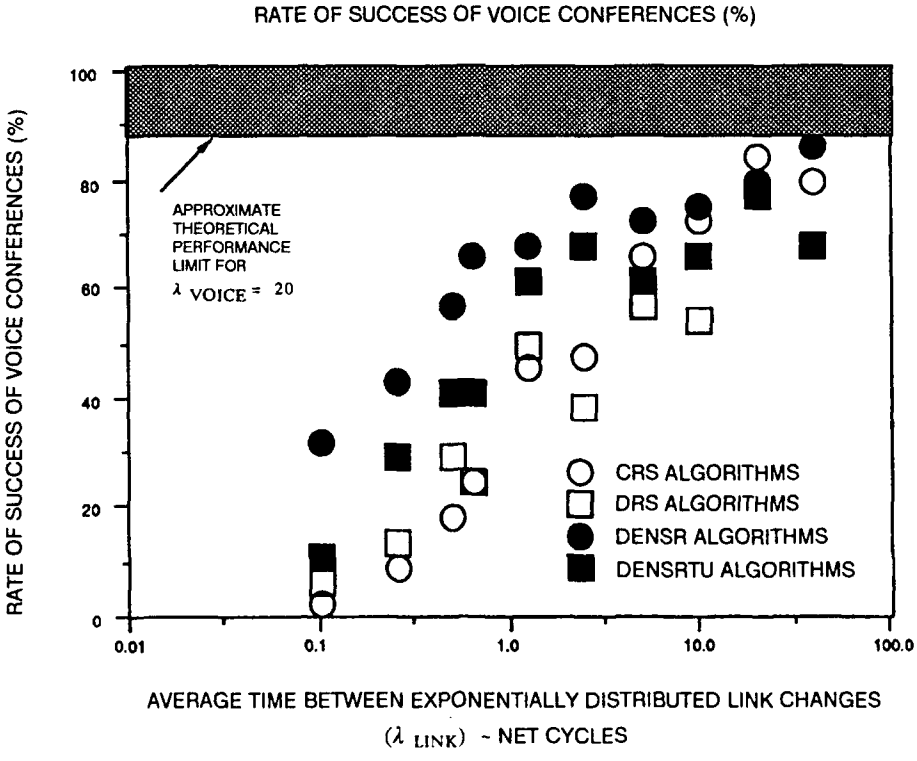


Figure 8. Graph of voice conferencing simulation results: 24-node network (figure 2a), one NTDS packet per node per 5 net cycles.

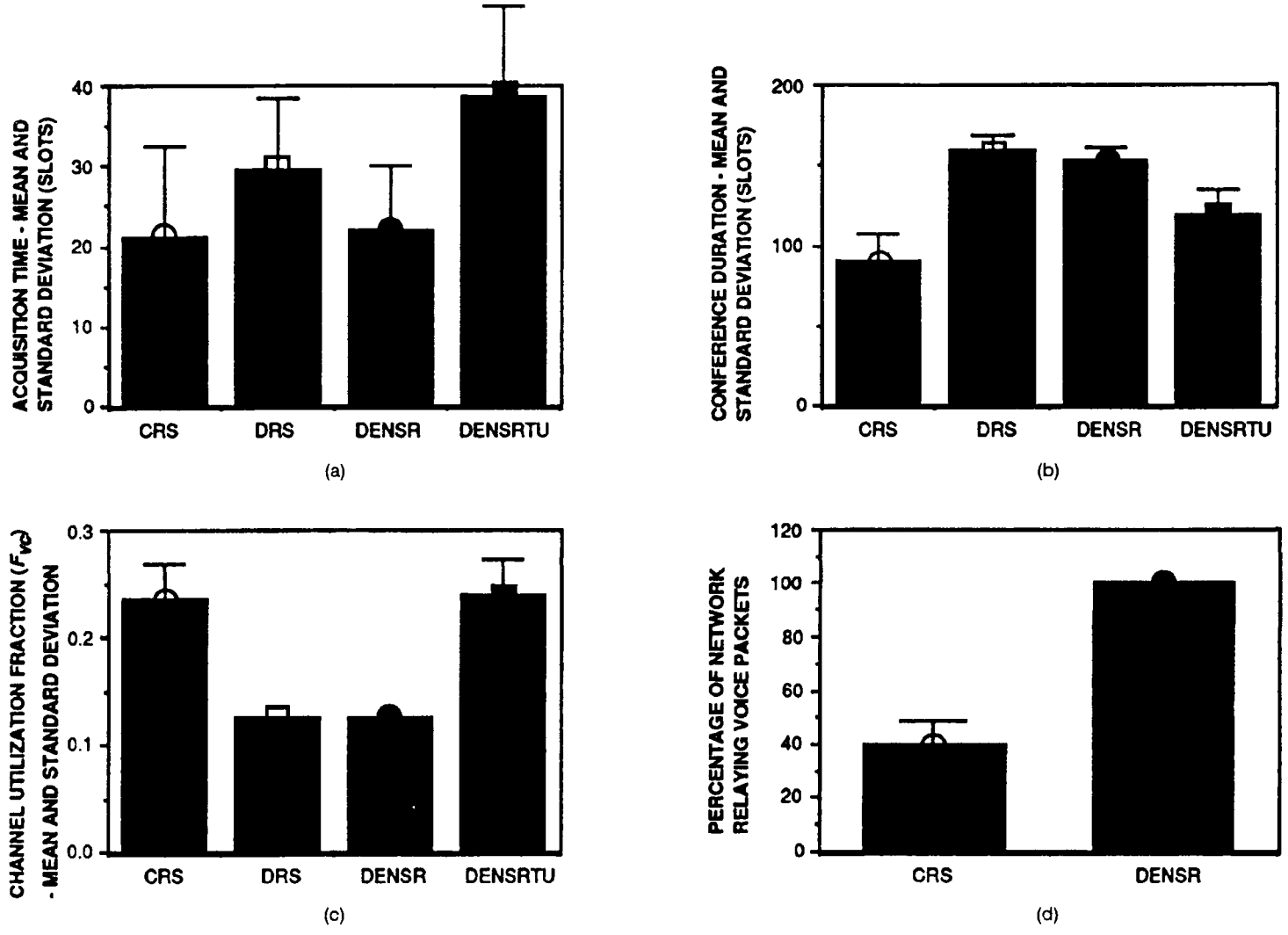


Figure 9. Additional statistics from voice conference simulation for 24-node network of figure 2a, under Model 1 of network topology dynamics. (a) Channel acquisition time. (b) Conference duration. (c) Channel utilization fraction (F_{vc}). (d) Percentage of network relaying voice packets. (Data are for simulations using one NTDS packet per net cycle; data for some algorithms not gathered in (d).)

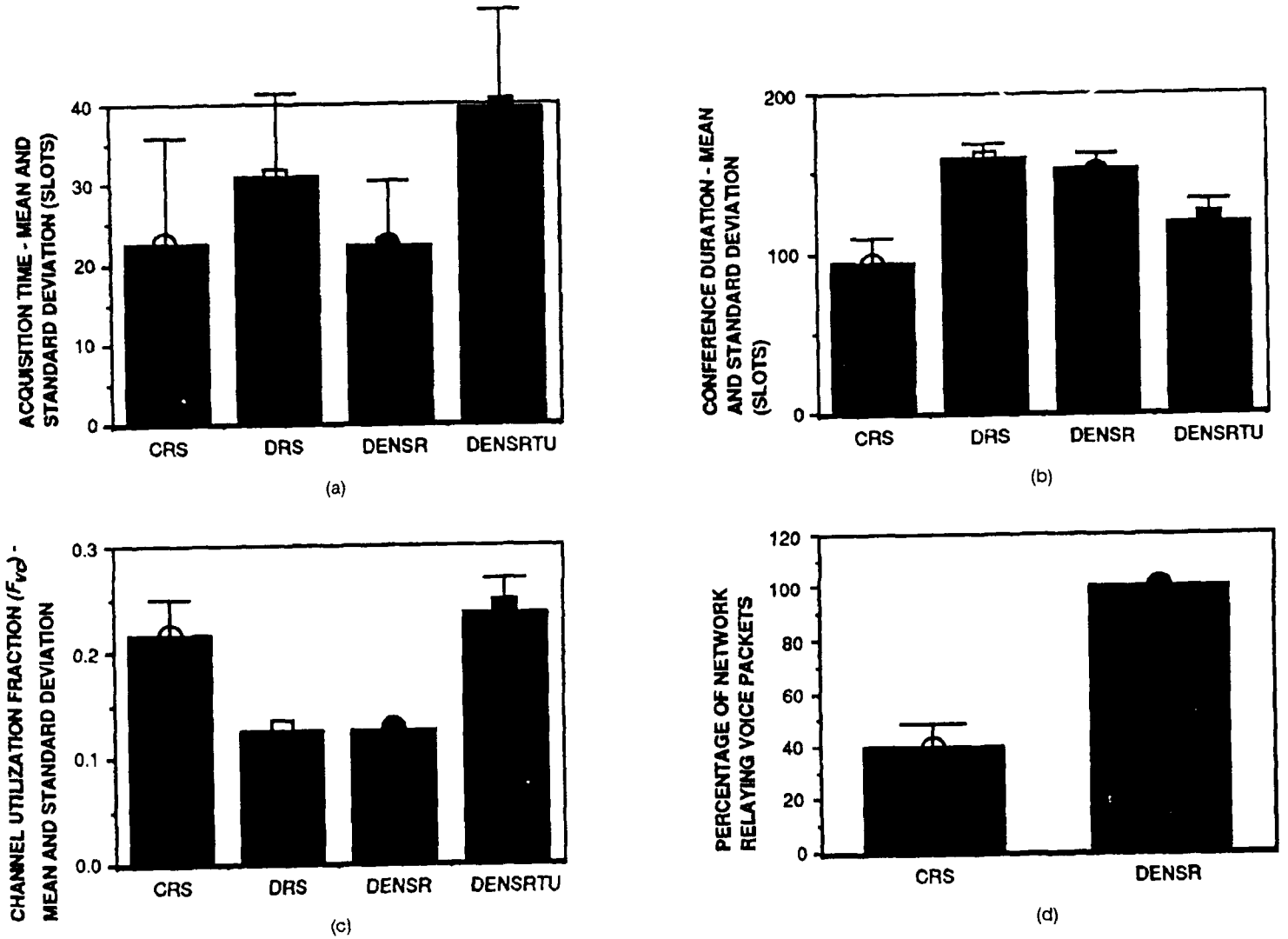


Figure 10. Additional statistics from voice conference simulation for 24-node network of figure 2a. (a) Channel acquisition time. (b) Conference duration. (c) Channel utilization fraction (F_{vd}). (d) Percentage of network relaying voice packets. (Data are for simulations using one NTDS packet per 5 net cycles; data for some algorithms not gathered in (d).)

$$P_{nocoll} = \left[\frac{T}{\lambda_{voice}}\right]^0 \frac{e^{-\frac{T}{\lambda_{voice}}}}{0!} = e^{-\frac{T}{\lambda_{voice}}}$$

where λ_{voice} is defined in the discussion leading to equation 6. Subtracting the above expression from unity gives the probability that one or more voice conference requests will arrive within the time T, or the probability P_{nocoll} that a conference will "collide" with the previous one:

$$P_{coll} = 1 - e^{-\frac{T}{\lambda_{voice}}} \tag{7}$$

Noting from the data plotted in figures 9 and 10 that voice conferences in these simulations have a total duration of between two to three net cycles, and using equation 6 for λ_{voice}, equation 7 shows that P_{coll} for these simulations is between 10 and 14%. This approximate result is used to determine the theoretical "ceiling" of maximum performance, which is shown by the horizontal line bounding the cross-hatched region at around 88% on the ordinates of figures 7 and 8. This can be viewed as the theoretical maximum performance line, assuming perfect knowledge of network topology, for the voice conference algorithms of these simulations.

Since the results of these simulations represent only a few (about 40-50) voice conference requests in simulations, which lasted about 1100 HAMA net cycles, and moreover since several of the events in the simulations are generated by random number generators, the results cannot be considered to be wholly quantitative, in the sense that running the simulations for a longer time or using different seeds for the random number generators would most likely result in slightly different statistics. Thus, the numerical validity and repeatability of the results may be questioned. Their value may be seen to lie more in pointing out which of the algorithms is most likely to give the most reliable generation of voice conference circuits and the routing and slotting errors to which each of the algorithms are prone, rather than actual quantitative accuracy. The ensuing discussion is offered in this vein, since a more complete statistical study that would have indicated error bars to be associated with each data point is beyond the scope of the current work.

The evident conclusion to be formed from figures 7 and 8 is that, on Model 1 of topology dynamics, the DENSR algorithm is the most successful in developing voice conference circuits. The DENSR algorithm exhibits around 80% success rate at all values of $\lambda_{link} >\approx 3$ net cycles on this model of topology dynamics and over 60% success at values down to $\lambda_{link} \approx 1$, with the "knee" of the DENSR performance curve occurring at $\lambda_{link} \approx 1$ net cycle. This algorithm continues to yield up to 30% success rate at values of $\lambda_{link} >\approx 0.1$ net cycles. The next most successful algorithm is the DENSRTU algorithm, which however usually does not exceed 70% performance for any values of λ_{link}. In comparison, both of the other algorithms tested—CRS and DRS—exhibit much

the same performance levels—about 60-70% success rates for values of $\lambda_{link} >\approx 8$ net cycles and 50% or less performance for values of $\lambda_{link} <\approx 5$ net cycles.

In summary, it is seen from figures 7 and 8 that under Model 1 of topology dynamics, the DENSR algorithm has fairly good performance down to values of λ_{link} less than one net cycle. This makes sense from the consideration that since the DENSR algorithm depends only on each node's knowledge of its neighbor set to make good slotting decisions, its performance should not degrade significantly unless this information is inaccurate. Obviously, inaccurate information can exist when the rate of link changes becomes more rapid than a node's ability to keep up with the changes. It requires one TDMA net cycle for all the slot-owner subslots to arise, through which, as observed earlier, each node becomes aware of the identities of its neighbors. Thus, if the average rate of link changes is significantly shorter than one net cycle, the DENSR algorithm would be expected to show degraded performance. Judging from figures 7 and 8, this characteristic appears to be substantiated, as the knee of the DENSR performance curve occurs at $\lambda_{link} \approx 1$.

The other algorithms that do not rely on the DENSR algorithm (i.e., the CRS and DRS algorithms) exhibit a rather continuous degradation of performance as λ_{link} is lowered. This must be attributed to their reliance on global network topology information, which becomes continually worse as the average rate of link changes increases. Thus, there is no noticeable "knee" in the performance curve of either of these algorithms, but instead a continual gradual decline as λ_{link} is lowered.

The penalty paid for using the DENSR algorithm is that voice conferences generally have longer duration than in the CRS algorithm because a larger number of voice slots per cycle are necessary to accommodate all the nodes in the network (figures 9 and 10). Also, as its name reminds us, the DENSR algorithm appropriates the entire network into the voice conference so no regular data transmissions may take place for the three net cycles or so the conference is in progress. However, the latter penalty should not be overestimated, as the CRS algorithm requires about 10 nodes to be involved in the conference for this 24-node network (figures 9 and 10), and if each of these nodes causes an additional node to have to remain silent or in passive mode, 80% of the network may be affected on the average by algorithms involving even the sparsest number of relays. It may be possible to lower the number of slots per voice cycle in the DENSR algorithm by having the caller node determine this number as equal to the degree of the network (according to its local topology) increased, for example, by two for safety. However, this variant has not been tested for Model 1 in this work, wherein as observed above a fixed number of eight slots per voice cycle was used for the DENSR algorithm. Dynamic determination of this number by the caller node could run the risk of too few slots being ordained for the current network requirements.

A study to determine the causes of failures, and their frequencies, under the DENSR algorithm for a relatively low value of $\lambda_{link} = 0.625$ (highly stressful network dynamics) was performed. The results of the study are summarized in figure 11, which shows that about 40% of the failures are due to corrupted CAP transmissions resulting from inconsistent CAP slotting assignments (CAP misslotting) on two disjoint slotting subgraphs. About 30% of the failures are due to collisions between circuits initiated within a short time of each other (see discussion of equation 7), about 15% are due to nodes not being aware of who their current neighbors are (locally erroneous topology information, see section 5), and the remainder are due to link changes occurring while the voice circuit was already functioning. Further analysis of the simulation results indicates that about one-third of the circuits that ran to "successful" completion at this value of λ_{link} nevertheless lost communication with one or more of their callees due to link changes during the time the circuit was functioning. These, however, were classified among the "successful" circuits since generally most of the callees received all voice messages. A successful voice conference under the DENSR algorithm is illustrated in figure 6, and an example of how a misslotting in the CAP slot assignments may cause a channel acquisition to fail is illustrated in figure 12. A successful voice conference under the DRS algorithms is illustrated in figure 13, while an unsuccessful conference resulting from CAP misslotting under the same algorithm is illustrated in figure 14.

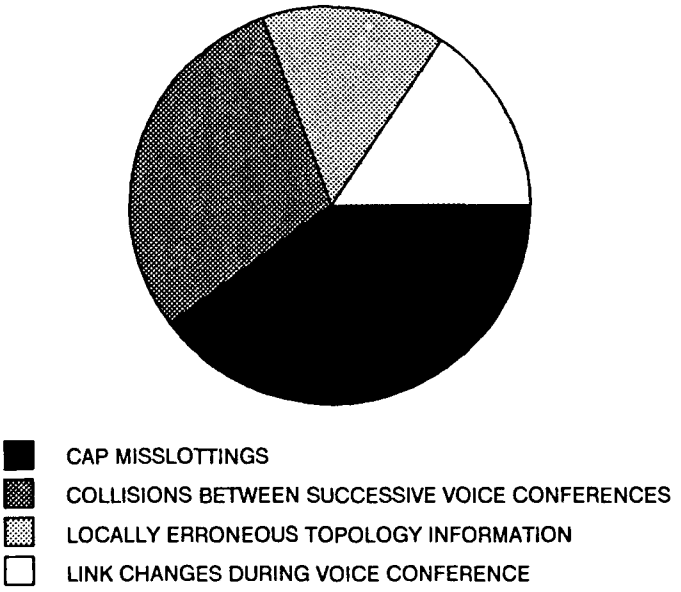


Figure 11. Analysis of main contributions to voice conference failures under DENSR algorithms with $\lambda_{link} = 0.625$.

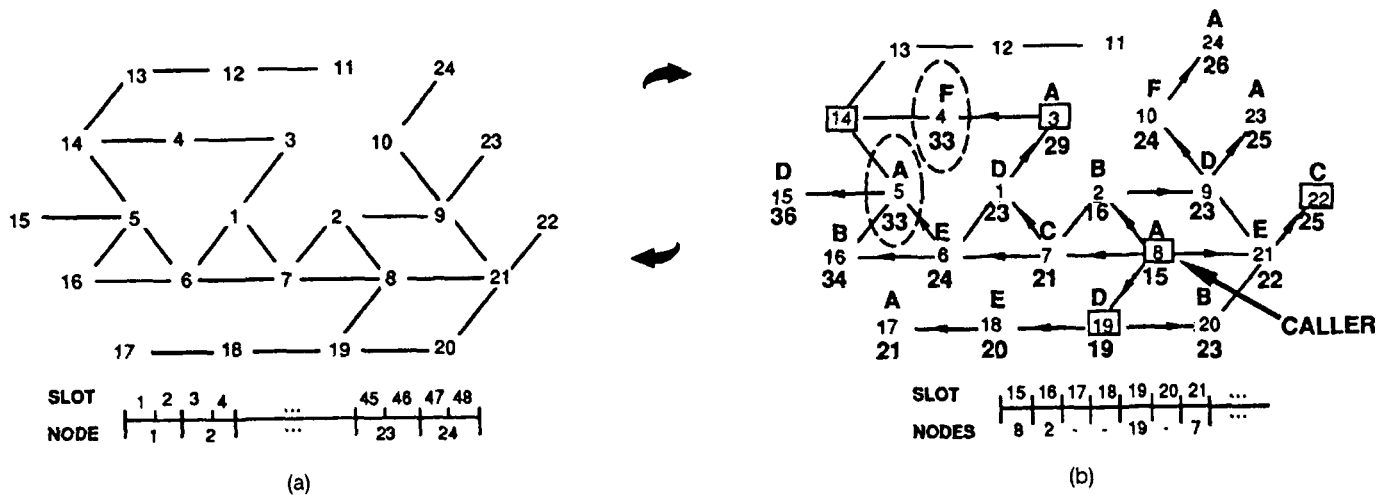


Figure 12. An unsuccessful attempt to acquire a voice channel under the DENSR algorithm. (a) Network under assigned slot TDMA protocol. (b) Node 8 (CALLER) attempting to acquire voice channel to callees denoted by boxes; attempt is unsuccessful because both nodes 4 and 5 (dashed ovals) have been assigned the same CAP slot (33), thus preventing node 14 from receiving an uncorrupted CAP. Network reverts back to HAMA.

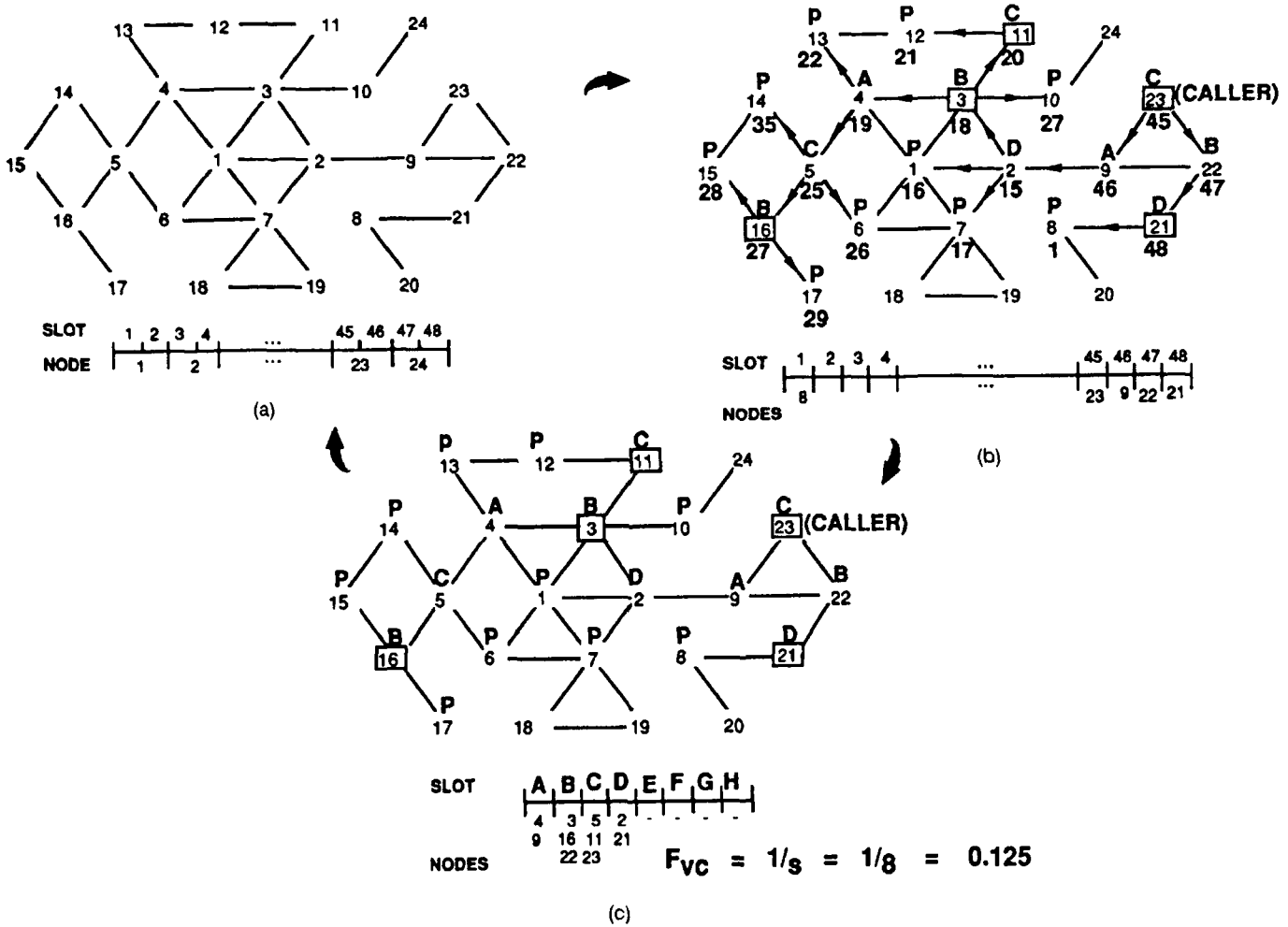


Figure 13. A successful voice conference under the DRS algorithm (caller and callees are in squares). (a) Network under HAMA. (b) Network showing CAP slotting (lower bold numbers), voice slotting (upper bold letters with P indicating passive nodes), and arrows showing path of CAPs. (c) Final slotting for voice conference (8 slot cycle). Network reverts to HAMA after end of conference.

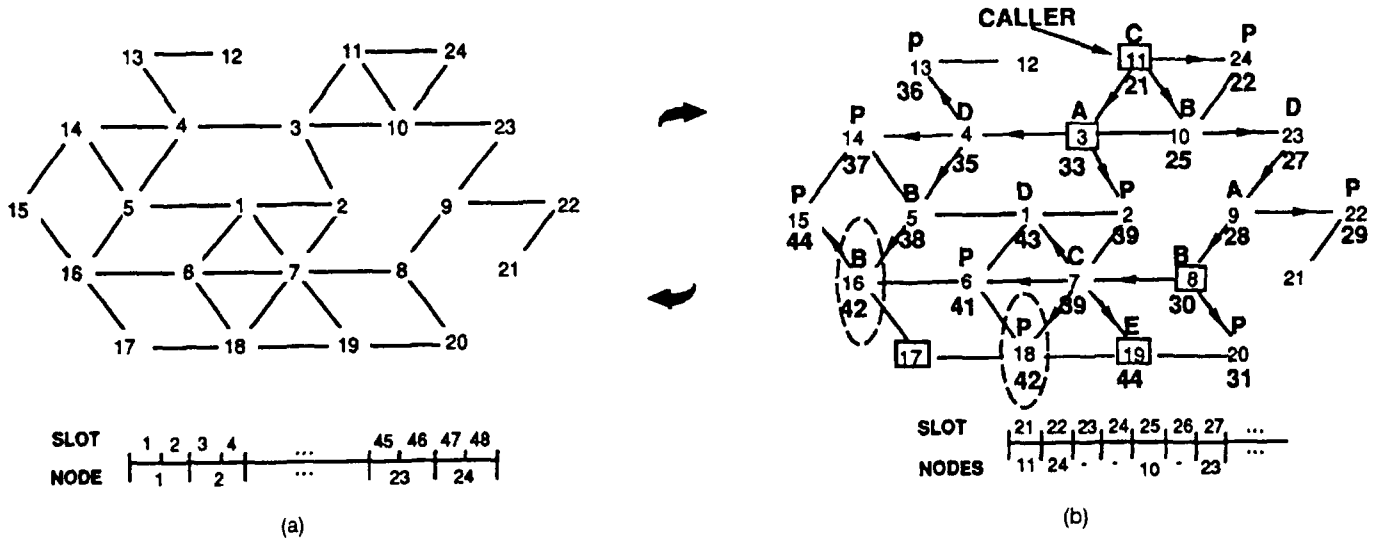


Figure 14. An unsuccessful voice conference under the DRS algorithm (caller and callees are in squares). (a) Network under HAMA. (b) Network showing conference members (squares), CAP slottings (lower bold numbers), and voice slottings (upper bold letters, with P indicating passive nodes). Nodes 16 and 18 in (b) have identical CAP slots, so callee 17 receives corrupted CAP and cannot be informed of voice conference request. Channel acquisition fails and network reverts back to HAMA protocol (a).

The DENSRTU algorithm is subject not only to above problems inherent in the DENSR algorithm (since initially it is identical to this algorithm), but it is more susceptible to link changes while the circuit is being set up or operational, on account of the selection of a unique path to each callee when setting up the circuit (using Djikstra's algorithm as outlined in section 3.4). This lack of redundancy in the DENSRTU algorithm results in degraded performance compared to the DENSR algorithm. It does, however, allow for fewer voice slots per cycle and fewer nodes participating in voice than the DENSR algorithm. Whether these advantages offset its degraded performance cannot be decided within the context of this study, although for tactical purposes high performance and reliability are at a premium.

The CRS and DRS algorithms exhibit about equivalent performance levels. This may be attributed to the fact that both these algorithms assume that nodes have accurate knowledge of the network topology, an assumption violated at low values of λ_{link}. The CRS algorithm, however, has advantages of economy in number of voice slots per cycle and number of nodes used in the voice circuit relative to the DENSR algorithm, as indicated in figures 8 and 9. Again, discussion of the relative advantages and disadvantages of these performance factors is beyond the scope of this work, although as above one expects tactical needs to favor high-reliability algorithms.

5.2 DISCUSSION OF SIMULATION OF TOPOLOGY DYNAMICS MODEL 2

*The performance of the four algorithms on Model 2 of topology dynamics is simu*lated in this work by allowing, for the purpose of simplicity, only two nodes—16 and 19 of figure 1—to move. As discussed earlier, in the course of their motion the nodes become neighbors of originally distant nodes and may create regions of high connectivity or high degree. With only two nodes moving, the degree of the network cannot exceed 9 (since it is 7 to begin with) so the required number of voice slots per cycle may be set equal to 10 for all algorithms except CRS. Moreover, the generation rate of NTDS packets for simulations on Model 2 is set equal to unity (the case of one NTDS packet per 5 net cycles is not simulated). The results of the simulation runs are shown in figure 15 for overall success rate of voice conferences (to be compared to figure 7 for performance under Model 1), and in figure 16 for other simulation statistics (analogous to figure 9 on Model 1).

We note from figure 15 the surprising result that the CRS algorithm, requiring as it does accurate global topology knowledge by each node, nevertheless gives the best performance on Model 2. The following factors may contribute to this result. First, the equivalent rate of link changes simulated under Model 2 is fairly low: the moving nodes move so slowly as to keep the topology invariant for several net cycles. Thus for comparison sake, the effective value of λ_{link} from Model 1 is around 5 and 10 net cycles. This permits nodes to "catch up" to current topology as messages are relayed

throughout the network, resulting in fairly intelligent routing and slotting decisions by the CRS algorithm. Moreover, as we have noted above, an important difference between Model 1 and Model 2 of topology dynamics is that on Model 2 the maximum degree of a node in the network can become as high as the number of nodes n in the network, if a sufficient number of nodes are allowed to change their positions. As the network on this model becomes strongly connected, the longest distance between any two nodes (radius) becomes low, possibly down to one to three hops. This contributes to improved cognizance by each node in the network of global topology, resulting in enhanced performance by the CRS algorithms. By the same token, it may render the additional overhead of the DENSR algorithms in terms of voice slots per cycle and time required to acquire the channel and complete the voice traffic, which is justified in cases wherein individual nodal topology information is inaccurate, no longer of benefit, and actually counterproductive. In a strongly connected network, algorithms of the DENSR class flooding as they do the entire network, may in principle require as many as n voice slots per cycle. Thus, they would yield no improvement in channel utilization fraction F_{vc} (see equation 4b) over the TDMA protocol, but would as a result of these low data rates require a long time to complete conference traffic.

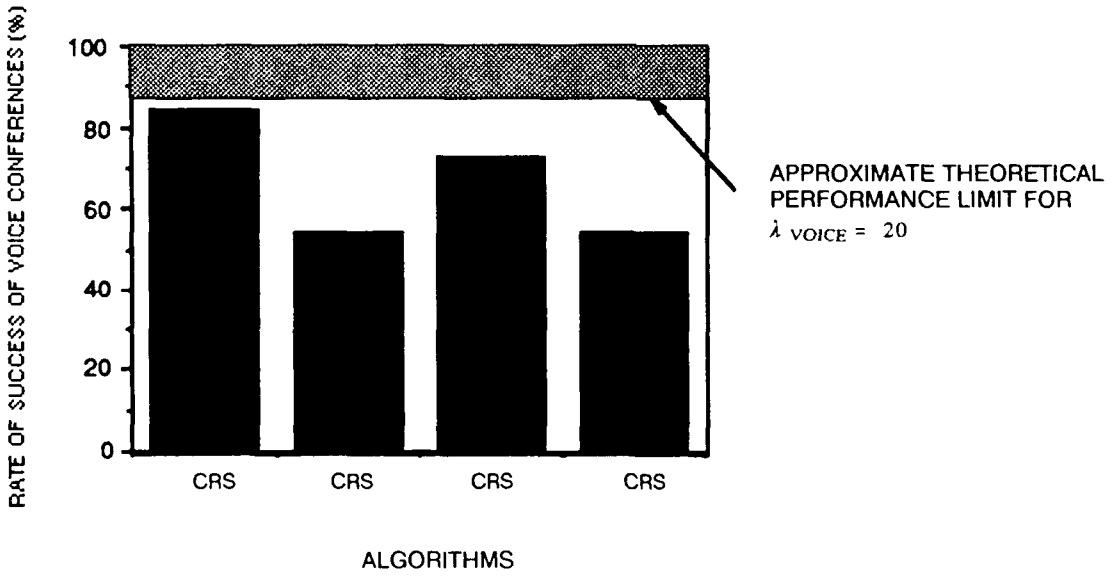


Figure 15. Voice conference success rates for the algorithms studied (Model 2 of network topology dynamics, 24-node network of figure 1a, one NTDS packet per node per net cycle.

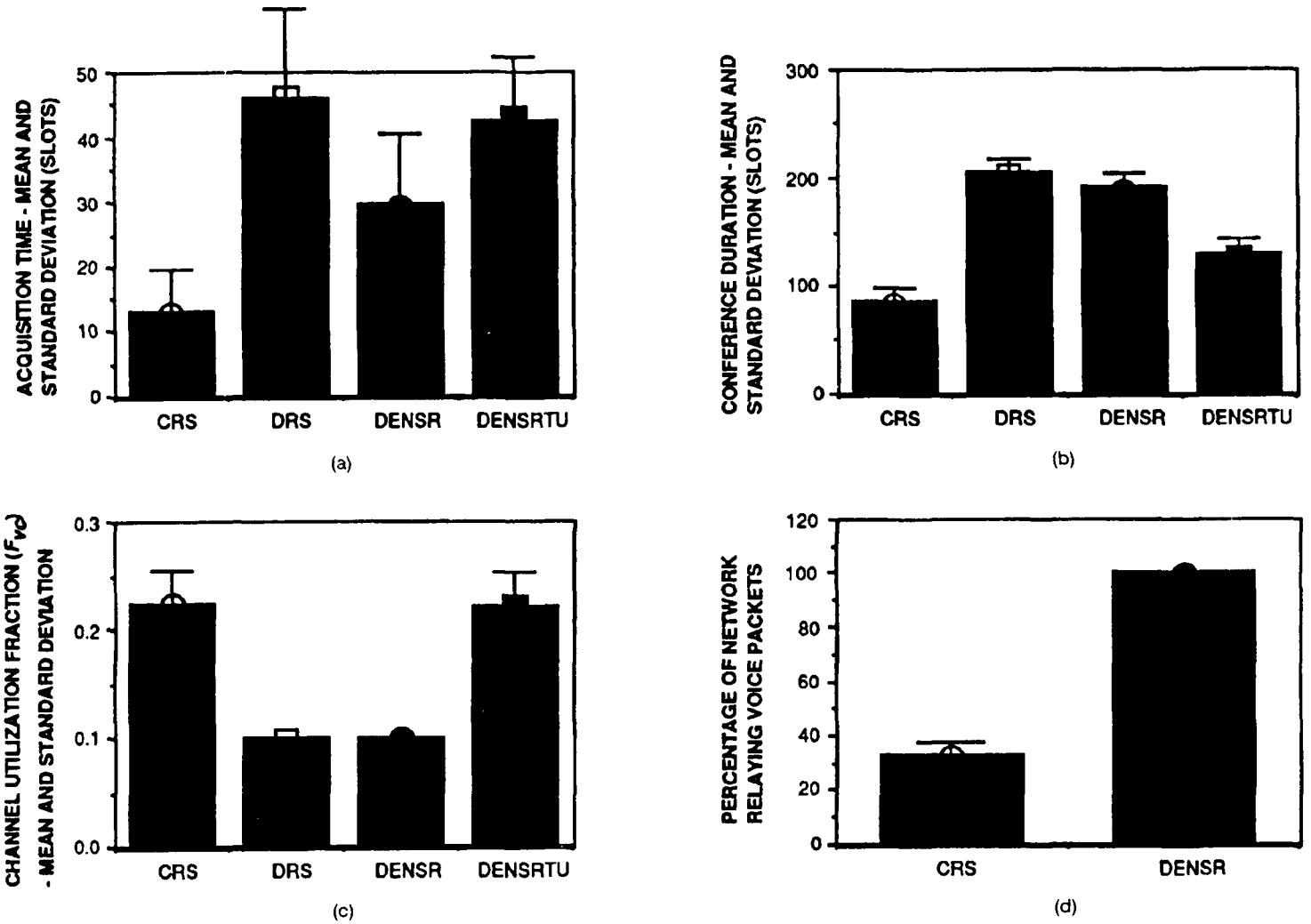


Figure 16. Additional statistics from voice conference simulation for 24-node network of figure 2a, under Model 2 of network topology dynamics. (a) Channel acquisition time. (b) Conference duration. (c) Channel utilization fraction (F_{vc}). (d) Percentage of network relaying voice packets. (Data are for simulations using one NTDS packet per net cycle; data for some algorithms not gathered in (d).)

The validity of this explanation may be partially substantiated by recalling (equation 7) that the longer the mean duration of a voice conference, the higher the probability P_{coll} that circuits will collide, resulting in failure. From figure 16, circuits on Model 2 using the CRS algorithms lasted about 1.8 net cycles, while circuits using the DRS and DENSR algorithms lasted about 4.2 net cycles, so that for the CRS algorithms

$$P_{coll,CRS} \approx .09$$

and for the DRS and DENSR algorithms,

$$P_{coll,DRS} \approx 0.18$$

$$P_{coll,DENSR} \approx 0.19$$

The roughly double collision probability for the DRS and DENSR algorithms compared to the CRS algorithms under Model 2 partly explains the poorer performance of the former. The DENSRTU algorithm was found to suffer an added degradation from such collision phenomena, for even though the duration of circuits under this algorithm is less than under the DENSR or DRS algorithms, it is found that topology updating sometimes actually results in poorer topology information being sent back to the caller compared to what that node already had. This is found to occur because some nodes have neighbors that had recently been passive in the previous voice circuit and the former nodes falsely sent back information that such passive nodes were no longer their neighbors (since they had not received transmissions from such passive nodes over the last net cycle).

Thus, under Model 2, the resulting network dynamics exhibit very low effective mean rates of link changes (on the order of 5 net cycles) when only two nodes are allowed to move relative to the rest of the network and the degree of the network graph can rise or fall within wide bounds. Under such a model, contrary to the findings on topology dynamics Model 1, a caller routed/time-slotted algorithm (CRS) can exhibit as good, if not better, performance than the other algorithms studied. On the other hand, if the degree of the network becomes very low (a condition not simulated in this work) one can expect the DENSR algorithm would outperform the others.

A successful voice conference under the CRS algorithm and on Model 2 of network dynamics is illustrated in figure 17, at a time when the two moving nodes 16 and 19 are close to one another.

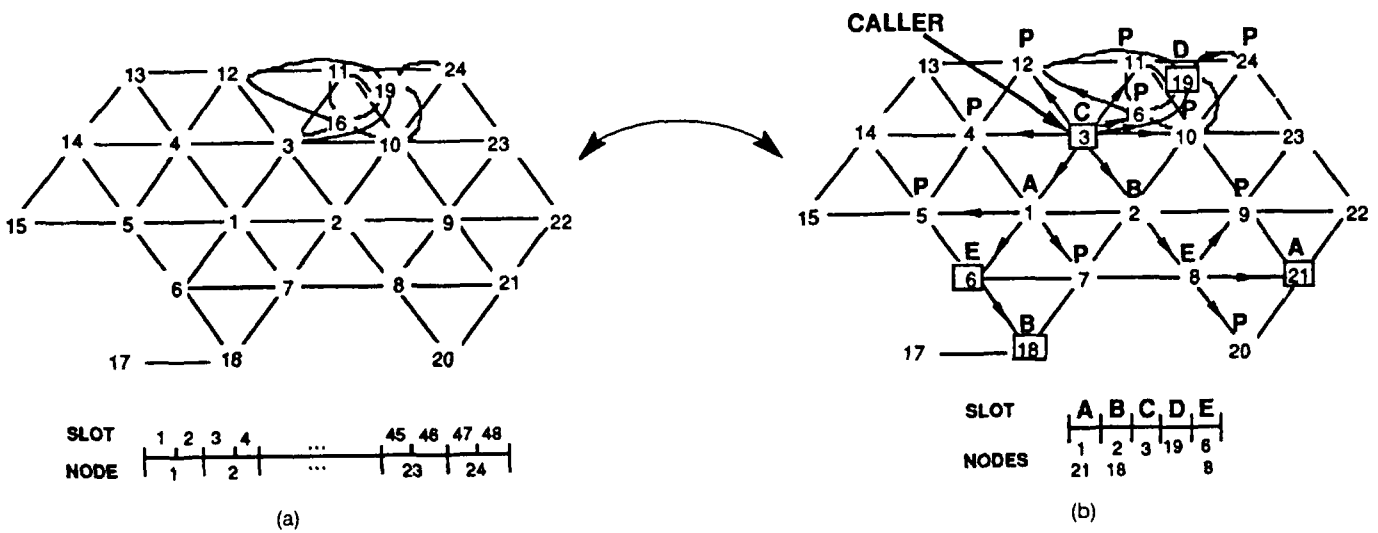


Figure 17. A successful voice conference on Model 2 of network dynamics and using the CRS algorithm. (a) Network under HAMA. (b) Reslotting of network under CRS algorithm (upper bold letters with P indicating passive nodes), conference members in squares. Time-slotting is not optimal, since 4 slots per cycle would have sufficed, however is functional. Network reverts to HAMA after termination of conference.

6.0 DISCUSSION AND CONCLUSIONS

The DENSR algorithm demonstrated the best performance of all the algorithms studied under Model 1 of topology dynamics (see figures 7 and 8). However, this algorithm has the drawbacks that it involves the entire network in voice traffic thus disabling it from data traffic and, additionally, it can require a large number (approaching n, the number of nodes in the network) of voice slots per cycle. This will happen when the network becomes highly connected, some evidence of which is seen in the simulation with topology dynamics Model 2 (see figures 16 and 17), wherein 10 voice slots per cycle were required, resulting in a channel utilization fraction of

$$F_{vc} = \frac{1}{10} \approx 0.1 . \tag{8}$$

This is only about a factor of 2.4 better than the value

$$F_{vc} = \frac{1}{24} \approx 0.04 \tag{9}$$

realized by a 24-node network under assigned slot TDMA. To propagate packetized voice with a minimum data rate of 1000 bps (see Gold, 1977) under the reslotting obeying equation 8 it would be necessary to have a physical (e.g., UHF-LOS) channel capable of supporting at least 10 kbps.

A secondary drawback of the DENSR algorithm is the necessity of the caller node knowing the degree of the network to appropriately set the number of voice slots per cycle. If this number is not accurately known by the caller node, this node may still make a reasonable choice by adding one or more to the maximum degree represented in its local version of the network topology. However, this form of the algorithm has not been tested in the simulation work reported herein.

If the network is highly connected at some nodes, the CRS algorithm can actually give slightly better performance than the DENSR algorithm (see figure 15), with much better economy in voice slots per cycle and in nodes participating in voice transmissions (figure 16). However, the DENSR algorithm still yields performance competitive with the CRS algorithm and can be considered overall the most reliable of the algorithms tested.

As a result of these tradeoffs between the DENSR and CRS algorithms, it may be advisable, for voice conferencing applications, to use a hybrid version of the CRS and DENSR algorithms, which allows choice of one or the other depending on the local information on the degree and radius of the network. For example, if the degree c of the network is expected to satisfy

$$c \approx n$$

and the radius is only three or four hops, the CRS algorithm would be considered the wiser choice based on its economy in F_{vc}, while if the degree of the network is expected to satisfy

$$c < \frac{n}{3}$$

and the radius is more than four hops, the DENSR algorithm would be preferred on account of its greater reliability over CRS and fair economy of F_{vc}. Thus, an intelligent decision-making process, which leads to a choice of the DENSR or the CRS algorithm, could create highly reliable voice circuits in a variety of topological realities.

It may be of value to simulate an embellishment of the CRS algorithm, in which any unsuccessful tries with this algorithm end by the contacted nodes sending back to the caller their local topology information, after which the caller makes a second (or third, etc.) try with the same algorithm but with expectedly improved topology information. Such a CRS algorithm with topology updating (which might be called CRSTU) has not been considered in this study, but might be an attractive alternative to the ones studied herein.

In this work, a distributed entire network slotting algorithm, with limited corrective reslotting capability (the DENSR algorithm), was developed for the purpose of packetized voice transmission. Dropping the constraints of voice propagation, one may choose instead to aim at the important goal of increasing D_{vc} of equation 4a for normal data (NTDS) transmissions in a tactical data network. An attractive approach to this general problem of distributed entire network time slotting/reslotting would be to begin the reslotting locally at the highest degree node(s) in the network, thus increasing the confidence that the chosen number of slots per cycle will be adequate. Moreover, the slot-owner subslots, which in the current study cease to exist during voice circuits, could be retained in their usual assigned slot TDMA mode to continually propagate local topology information, thus facilitating corrective action for misslottings. In this manner, it is conceivable that the network can operate with a continual economy of number of data slots per cycle, thereby significantly increasing the effective data rate per node, D_{vc}. This will be the subject of a follow-on study.

7.0 REFERENCES

Christofides, N. 1975. *Graph Theory—An Algorithmic Approach*. Academic Press, New York, pp. 7, 14, 58-78, chapter 4.

Gold, B. 1977. "Digital Speech Networks." *Proc. IEEE*, v. 65, n. 12, Dec., pp. 1636-1658.

Gutman, L., R. Casey, R. Merk, D. Olsen, and C. Warner. 1988. "UHF-LOS Network Architecture Specification for the Unified Networking Technology (UNT) Program," UNT Program Office, Report No. 250-U-0002, pp. 47-52, pp. 131-148.

Merk, R., L. Gutman, and C. Warner. 1988. "Voice and Data Integration in Tactical Networks," *IEEE Military Communications Conference*, v. 3, pp. 1069-1073, San Diego, CA, Oct. 23-26.

Mumm, H. and R. Ollerton. 1990. "User's Guide to an Event-Activation Record Approach to Simulation Modeling in ADA," Naval Ocean Systems Center, San Diego, NOSC TD 1944 (Dec).

Norvell, S., G. J. Brown and M. Vineberg. 1985. "Modeling and Evaluation of Multiple Access Protocols," *1985 IEEE Military Communications Conference* (MILCOM '85), sponsored by IEEE Communications Society, U.S. Department of Defense, Armed Forces Communications and Electronics Association, October 20-23, Boston, MA., pp. 5.2.1–5.2.5.

Tannenbaum, A. S. 1988. *Computer Networks*, Prentice-Hall, New Jersey, pp. 84-86.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	January 1991	Final

4. TITLE AND SUBTITLE

GRAPH COLORING APPLIED TO VOICE CONFERENCING IN TACTICAL PACKET RADIO NETWORKS

5. FUNDING NUMBERS

0601152N
RR000N0
ZW30
DN300004

6. AUTHOR(S)

N. Davé

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Ocean Systems Center
San Diego, CA 92152–5000

8. PERFORMING ORGANIZATION REPORT NUMBER

NOSC TD 2036

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Office of Chief of Naval Research
OCNR–10P
Arlington, VA 22217–5000

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

In the course of the work reported herein, a number of algorithms are developed to establish temporarily higher data rate voice communications in an ultrahigh frequency line-of-sight (UHF-LOS) network normally obeying assigned time division multiple access (TDMA) or handoff assigned multiple access (HAMA) channel access protocols. These algorithms are based on graph coloring theory. They are tested in a discrete event simulation testbed under two different models of network topology dynamics. The algorithms are compared based on simulation results using the criteria of success rate of voice conference circuits, effective data rate or slots per cycle, percentage of network involved in voice traffic, and duration of voice circuits. Conclusions regarding the most favorable algorithm under different network connectivity realities are reached. Possible hybridized algorithms, involving a combination of the algorithms studied, the algorithm chosen depending on the local perception of network connectivity, are suggested. A continually evolving, economical slot assignment algorithm (reslotting) of dynamic TDMA networks is proposed for future study.

14. SUBJECT TERMS

graph coloring
network reslotting
packet radio networks

15. NUMBER OF PAGES

58

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

21a. NAME OF RESPONSIBLE INDIVIDUAL	21b. TELEPHONE *(include Area Code)*	21c. OFFICE SYMBOL
N. Davé	(619) 553-4967	Code 854

INITIAL DISTRIBUTION

Code 0012	Patent Counsel	(1)
Code 0141	Dr. A. Gordon	(2)
Code 0144	R. November	(1)
Code 80	K. D. Regan	(1)
Code 85	D. R. Casey	(1)
Code 8503	Dr. C. J. Warner	(1)
Code 8503	G. J. Brown	(1)
Code 8503	S. Nowell	(1)
Code 8503	A. Heaberlin	(1)
Code 854	R. L. Merk	(1)
Code 854	N. Dave'	(5)
Code 952B	J. Puleo	(1)
Code 961	Archive/Stock	(6)
Code 964B	Library	(3)

Defense Technical Information Center
Alexandria, VA 22304-6145 (4)

NOSC Liaison Office
Washington, DC 20363-5100 (1)

Center for Naval Analyses
Alexandria, VA 22302-0268 (1)